

# Hands-On Lab: NAV on Docker

## Who should complete this HOL?

This Hands-On Lab is designed to help you understand what Docker is and what NAV on Docker can do for you. After completing the HOL, you should be able to determine if Docker and especially NAV on Docker is useful in your organization. The HOL will use the Workshop VMs as a foundation for the HOL to have a uniform platform for all.

When you have completed this HOL, you can find more info on the nav-docker project on github: <http://www.github.com/microsoft/nav-docker>. This is also the place you should be filing issues and comments.

## What is Docker?

If you are new to Docker and Containers, you might want to scan through this document before heading into the workshop:

<https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/>

This should give you a better understanding of what Docker is.

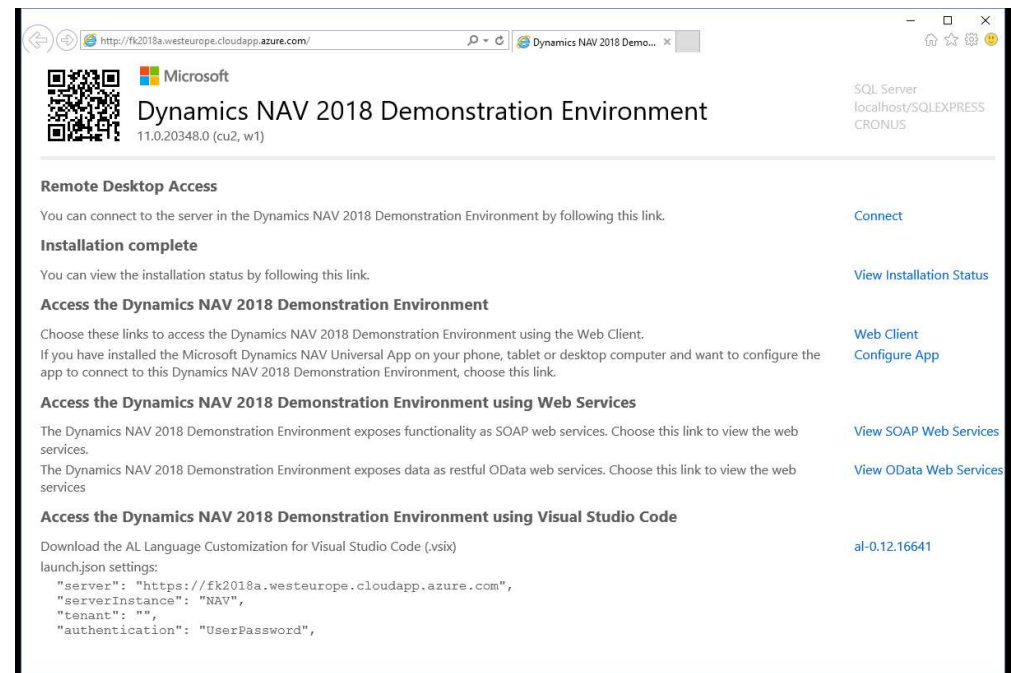
For the remaining of the workshop, you will be going through some scenarios, using Docker, on how to Deploy NAV. For completing these scenarios, you will need a learning environment, which you can get using <http://aka.ms/getnav> if you are completing the workshop at home.

When you connect to your learning environment, you are presented with a website, which looks like the image on the right side. What you might not be aware is, that when you are viewing this, you are already using Docker. This website is hosted on an Azure Virtual Machine, but NAV is not installed on the VM. The VM is a Docker host and NAV is running in a Container on the host.

Clicking the Connect link will download the .rdp file, which launches Remote Desktop to the Workshop VM.

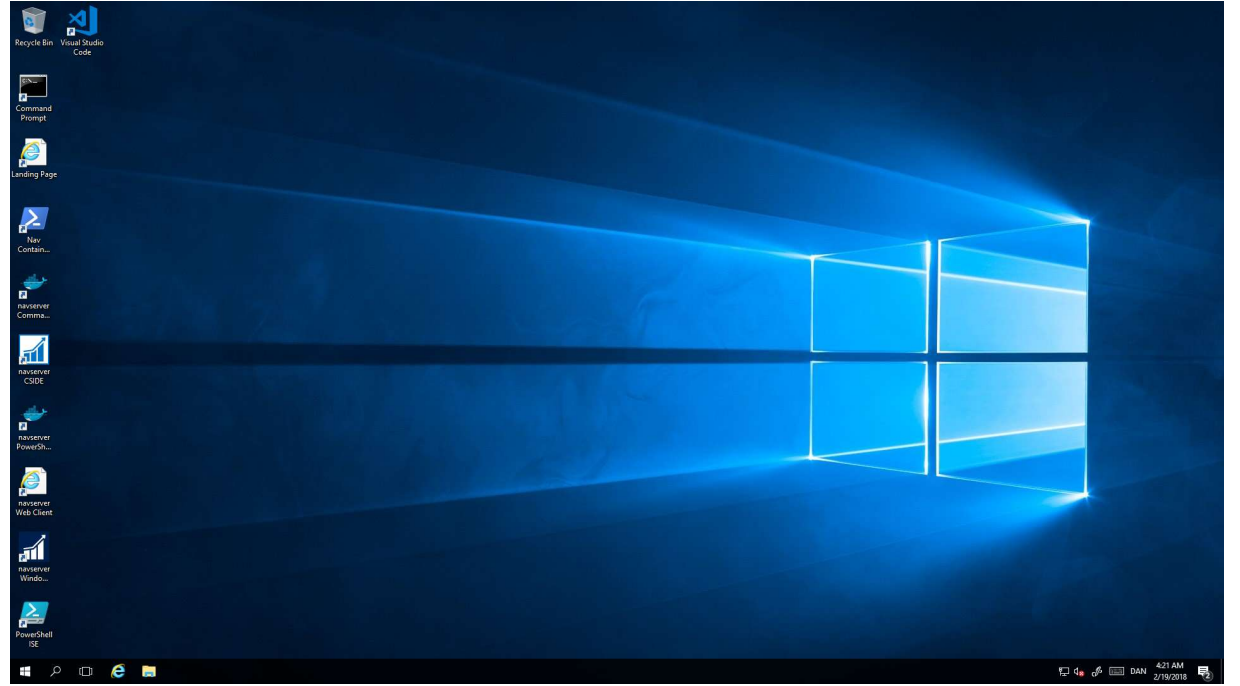
Note, that you can connect to the Docker host (the Azure Virtual Machine), but you will not and cannot connect to a remote desktop in the container itself. The Container is based on WindowsServerCore, which has no UI, no desktop.

Connecting to the Workshop VM (the Docker host) will allow you to interact with the Docker Containers that are available on that machine by using various commands.



First thing we will do is to have a look at the Workshop VM desktop and what we can do with that.

Open your workshop landing page in a browser and press the Connect button to connect to the remote desktop of your workshop environment.  
Use the credentials provided to login.

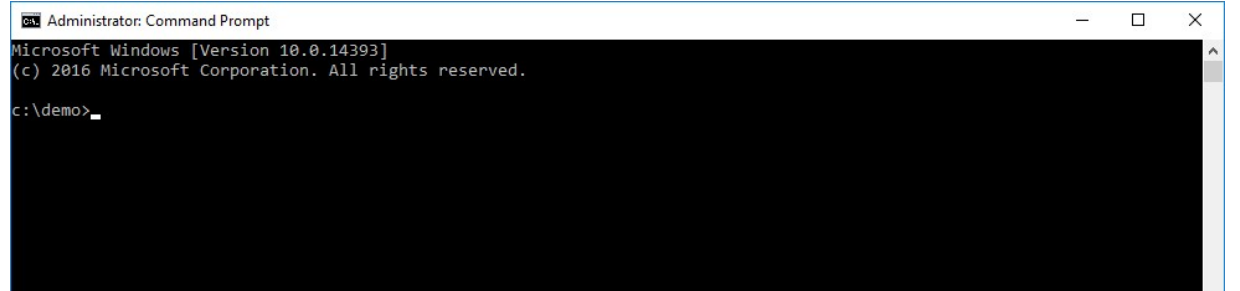


### Command Prompt

The Command Prompt is the standard CMD.EXE running as Administrator.

This prompt is primarily there for running Docker commands or other executables.

We will be using the Command Prompt throughout this Hands On Lab.



## Nav Container Helper

The navcontainerhelper is a set of functions, which will help you working with Nav Containers.

When you start the container helper, it will display a number of the available functions.

**Note** that the navcontainerhelper is an open source project from <http://www.github.com/microsoft/navcontainerhelper> and any issues regarding the navcontainerhelper should be added under issues in the github repo.

We will dive into the container helper later.



```
Administrator: Nav Container Helper
Welcome to the Nav Container Helper PowerShell Prompt

Container info functions
Get-NavContainerNavVersion      Get Nav version from NAV container or image
Get-NavContainerImageName      Get ImageName from NAV container
Get-NavContainerGenericTag     Get Nav generic image tag from NAV container or image
Get-NavContainerOsVersion      Get OS version from NAV container or image
Get-NavContainerEula           Get Eula link from NAV container or image
Get-NavContainerLegal          Get Legal link from NAV container or image
Get-NavContainerCountry        Get country version from NAV container or image
Get-NavContainerIpAddress      Get IP Address to a NAV container
Get-NavContainerSharedFolders  Get Shared Folders from a NAV container
Get-NavContainerPath           Get the path inside a NAV container to a shared file
Get-NavContainerName           Get the name of a NAV container
Get-NavContainerId             Get the Id of a NAV container
Test-NavContainer              Test whether a NAV container exists
Get-NavContainerDebugInfo      Get Troubleshooting info for NAV container if you need help with an issue
Get-NavContainers.ps1         Get All Nav Containers
Get-NavContainerEventLog.ps1   Get EventLog from Nav Container

Container handling functions
New-NavContainer               Create new Nav container
New-CSideDevContainer          Create new C/SIDE development container
Remove-NavContainer            Remove Nav container
Get-NavContainerSession        Create new session to a Nav container
Remove-NavContainerSession     Remove Nav container session
Enter-NavContainer             Enter Nav container session
Open-NavContainer              Open Nav container in new window
Wait-NavContainerReady         Wait for Nav Container to become ready
Copy-FileFromNavContainer      Copy file from Nav Container
Copy-FileToNavContainer        Copy file to Nav Container
Export-NavContainerDatabasesAsBacpac Export database(s) in Nav Container as BacPac

Object handling functions
Import-ObjectsToNavContainer   Import objects from .txt or .fob file to Nav Container
Import-DeltasToNavContainer     Merge delta files and Import objects to Nav Container
Import-TestToolkitToNavContainer Import TestToolkit to Nav Container
Compile-ObjectsInNavContainer  Compile objects
Export-NavContainerObjects     Export objects from Nav container
Create-MyOriginalFolder        Create folder with the original objects for modified objects
Create-MyDeltaFolder           Create folder with deltas for modified objects
Convert-Txt2Al                 Convert deltas folder to al folder
Export-ModifiedObjectsAsDeltas Export objects, create baseline and create deltas
Convert-ModifiedObjectsToAl    Export objects, create baseline, create deltas and convert to .al files

App handling functions
Publish-NavContainerApp        Publish App to Nav container
Sync-NavContainerApp           Sync App in Nav container
Install-NavContainerApp         Install App in Nav container
Uninstall-NavContainerApp      Uninstall App from Nav container
Unpublish-NavContainerApp      Unpublish App from Nav container
Get-NavContainerAppInfo        Get info about installed apps from Nav Container
Start-NavContainerAppDataUpgrade Start Data Upgrade for an App in a Nav Container
Install-NAVSipCryptoProviderFromNavContainer Install Nav Sip Crypto Provider locally from container to sign extensions

Tenant handling functions
New-NavContainerTenant         Create tenant in multitenant Nav Container
Remove-NavContainerTenant      Remove tenant from multitenant Nav Container

User handling functions
New-NavContainerNavUser        Create new Nav User in Nav Container
New-NavContainerWindowsUser    Create new Windows User in Nav Container
```



### navserver Command Prompt

The navserver Command Prompt is the standard CMD.EXE running inside the navserver container.

When you run **dir** inside the navserver Command Prompt you will see the Container file system. Folders that are shared from the host to the container are shown as symbolic directory links (SYMLINKD).

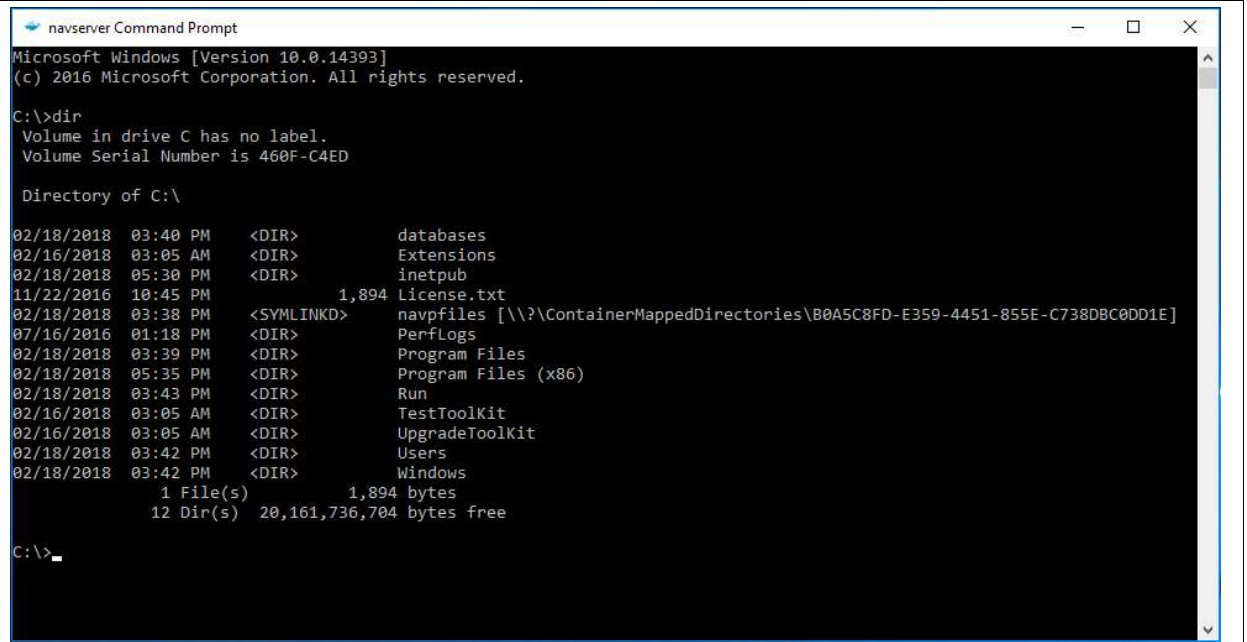
The file system inside the NAV Docker Image consists of a few special folders/files:

**c:\run** the run folder is the folder containing all the scripts, which are used to set up NAV in the container.

**c:\run\my** is the location, where you can place scripts which can override functionality of the run folder. Typical scenario is to share a folder from the host to the c:\run\my folder, containing various scripts that you want executed during start.

**c:\run\start.ps1** is the entry point for the container.

**c:\run\navstart.ps1** is the main script for setting up NAV and launching other setup scripts.



```
navserver Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\>dir
Volume in drive C has no label.
Volume Serial Number is 460F-C4ED

Directory of C:\

02/18/2018  03:40 PM  <DIR>          databases
02/16/2018  03:05 AM  <DIR>          Extensions
02/18/2018  05:30 PM  <DIR>          inetpub
11/22/2016  10:45 PM             1,894 License.txt
02/18/2018  03:38 PM  <SYMLINKD>    navpfiles [\\?\ContainerMappedDirectories\B0A5C8FD-E359-4451-855E-C738DBC0DD1E]
07/16/2016  01:18 PM  <DIR>          PerfLogs
02/18/2018  03:39 PM  <DIR>          Program Files
02/18/2018  05:35 PM  <DIR>          Program Files (x86)
02/18/2018  03:43 PM  <DIR>          Run
02/16/2018  03:05 AM  <DIR>          TestToolKit
02/16/2018  03:05 AM  <DIR>          UpgrndeToolKit
02/18/2018  03:42 PM  <DIR>          Users
02/18/2018  03:42 PM  <DIR>          Windows
               1 File(s)              1,894 bytes
               12 Dir(s)      20,161,736,704 bytes free

C:\>
```

### navserver PowerShell Prompt

The navserver PowerShell Prompt is a PowerShell prompt running inside the container.

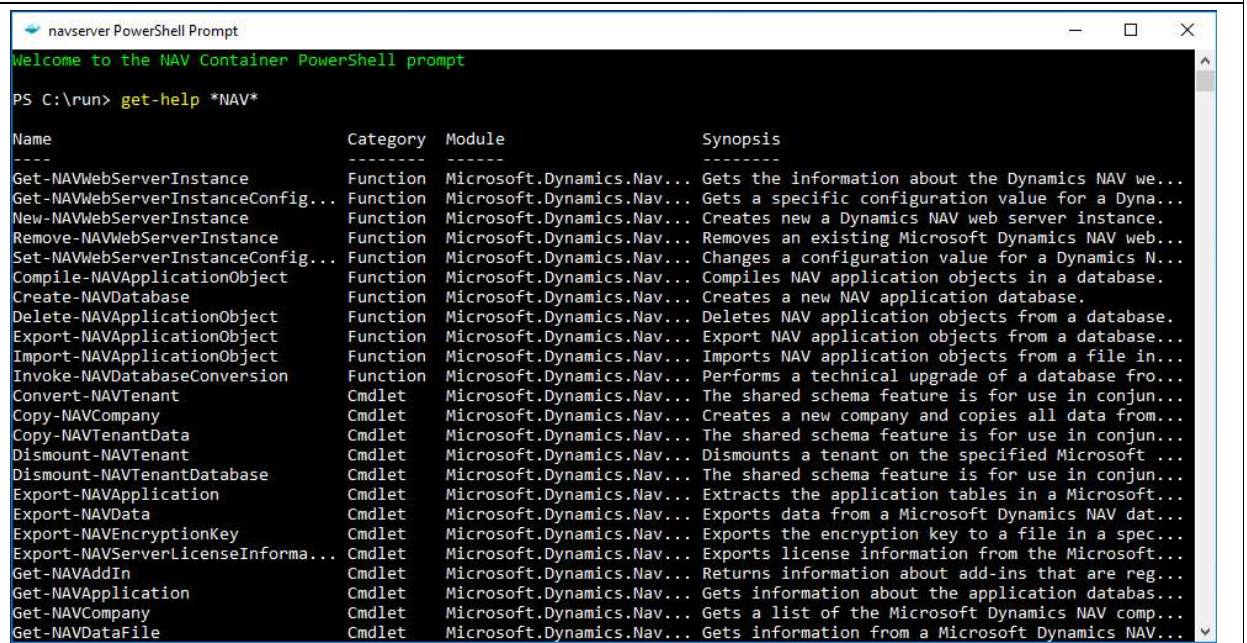
All NAV cmdlets are loaded inside the PowerShell prompt, ready to use.

Example:

#### Get-NavServerUser NAV

Will list all users in the NAV server instance (which is the default server instance in the container).

Note that **not** all commands will work inside the container. You cannot create a new server instance, for example – that is done by spinning up another container (the Docker way 😊)



```
navserver PowerShell Prompt
Welcome to the NAV Container PowerShell prompt

PS C:\run> get-help *NAV*

Name                Category  Module  Synopsis
----                -
Get-NAVWebServerInstance  Function Microsoft.Dynamics.Nav... Gets the information about the Dynamics NAV we...
Get-NAVWebServerInstanceConfig... Function Microsoft.Dynamics.Nav... Gets a specific configuration value for a Dyna...
New-NAVWebServerInstance  Function Microsoft.Dynamics.Nav... Creates new a Dynamics NAV web server instance.
Remove-NAVWebServerInstance  Function Microsoft.Dynamics.Nav... Removes an existing Microsoft Dynamics NAV web...
Set-NAVWebServerInstanceConfig... Function Microsoft.Dynamics.Nav... Changes a configuration value for a Dynamics N...
Compile-NAVApplicationObject  Function Microsoft.Dynamics.Nav... Compiles NAV application objects in a database.
Create-NAVDatabase         Function Microsoft.Dynamics.Nav... Creates a new NAV application database.
Delete-NAVApplicationObject  Function Microsoft.Dynamics.Nav... Deletes NAV application objects from a database.
Export-NAVApplicationObject  Function Microsoft.Dynamics.Nav... Export NAV application objects from a database...
Import-NAVApplicationObject  Function Microsoft.Dynamics.Nav... Imports NAV application objects from a file in...
Invoke-NAVDatabaseConversion  Function Microsoft.Dynamics.Nav... Performs a technical upgrade of a database fro...
Convert-NAVTenant          Cmdlet   Microsoft.Dynamics.Nav... The shared schema feature is for use in conjun...
Copy-NAVCompany            Cmdlet   Microsoft.Dynamics.Nav... Creates a new company and copies all data from...
Copy-NAVTenantData         Cmdlet   Microsoft.Dynamics.Nav... The shared schema feature is for use in conjun...
Dismount-NAVTenant         Cmdlet   Microsoft.Dynamics.Nav... Dismounts a tenant on the specified Microsoft ...
Dismount-NAVTenantDatabase  Cmdlet   Microsoft.Dynamics.Nav... The shared schema feature is for use in conjun...
Export-NAVApplication       Cmdlet   Microsoft.Dynamics.Nav... Extracts the application tables in a Microsoft...
Export-NAVData              Cmdlet   Microsoft.Dynamics.Nav... Exports data from a Microsoft Dynamics NAV dat...
Export-NAVEncryptionKey     Cmdlet   Microsoft.Dynamics.Nav... Exports the encryption key to a file in a spec...
Export-NAVServerLicenseInforma... Cmdlet   Microsoft.Dynamics.Nav... Exports license information from the Microsoft...
Get-NAVAddIn                Cmdlet   Microsoft.Dynamics.Nav... Returns information about add-ins that are reg...
Get-NAVApplication         Cmdlet   Microsoft.Dynamics.Nav... Gets information about the application databas...
Get-NAVCompany              Cmdlet   Microsoft.Dynamics.Nav... Gets a list of the Microsoft Dynamics NAV comp...
Get-NAVDataFile             Cmdlet   Microsoft.Dynamics.Nav... Gets information from a Microsoft Dynamics NAV...
```

### navserver CSIDE

C/SIDE a.k.a. the Classic Development Environment for the navserver container.

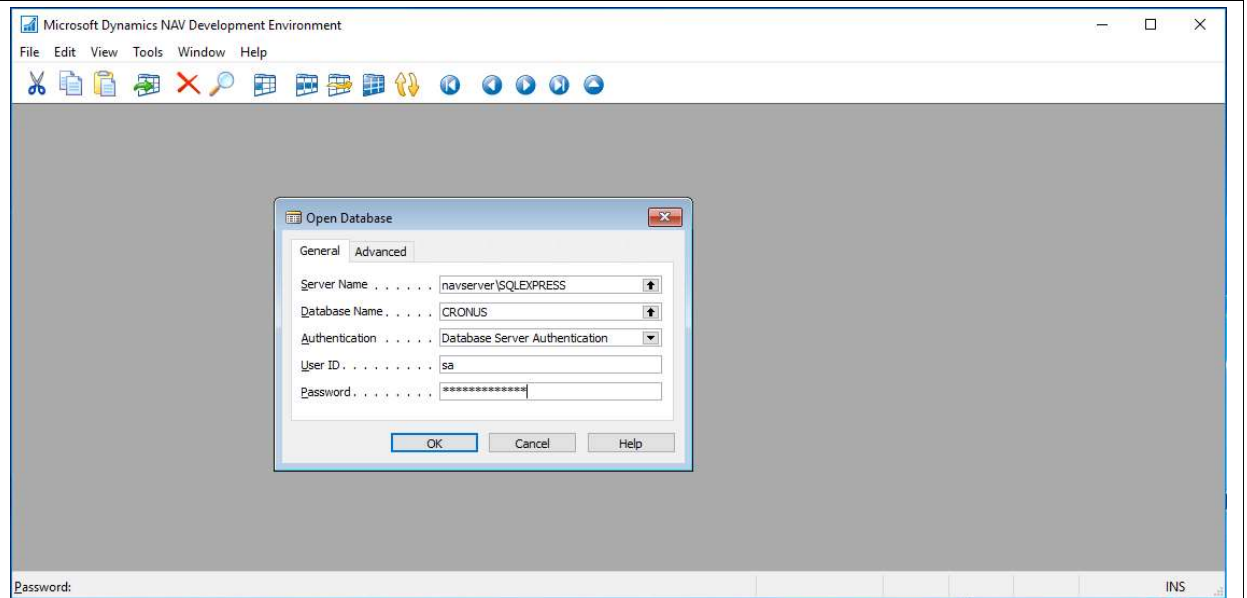
Note that C/SIDE is not there to support all classic development scenarios.

The primary reason for C/SIDE to be available is for the VS Code developer to be able to see and browse through the source of the base application.

Having said that, you can do the majority of classic development scenarios in C/SIDE.

Note that when you start C/SIDE you will be running Database Authentication and you have to login as **SA** and use the Workshop VM password.

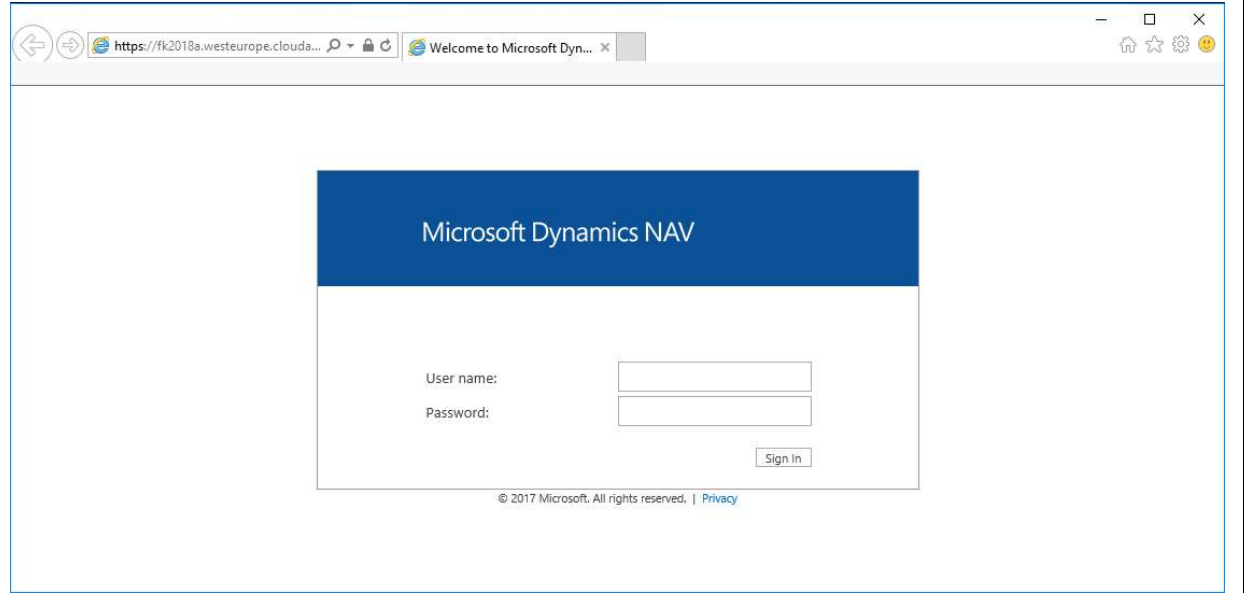
Server name is **navserver\SQLEXPRESS** and the database name depends on which localization you are running.



### navserver Web Client

Opens a browser with the Web client for the navserver container. The Web client is installed inside the container on IIS and the ports are exposed on the container and published to the host.

Use the credentials provided to login.



### navserver Windows Client

Opens the Windows client for the navservercontainer.

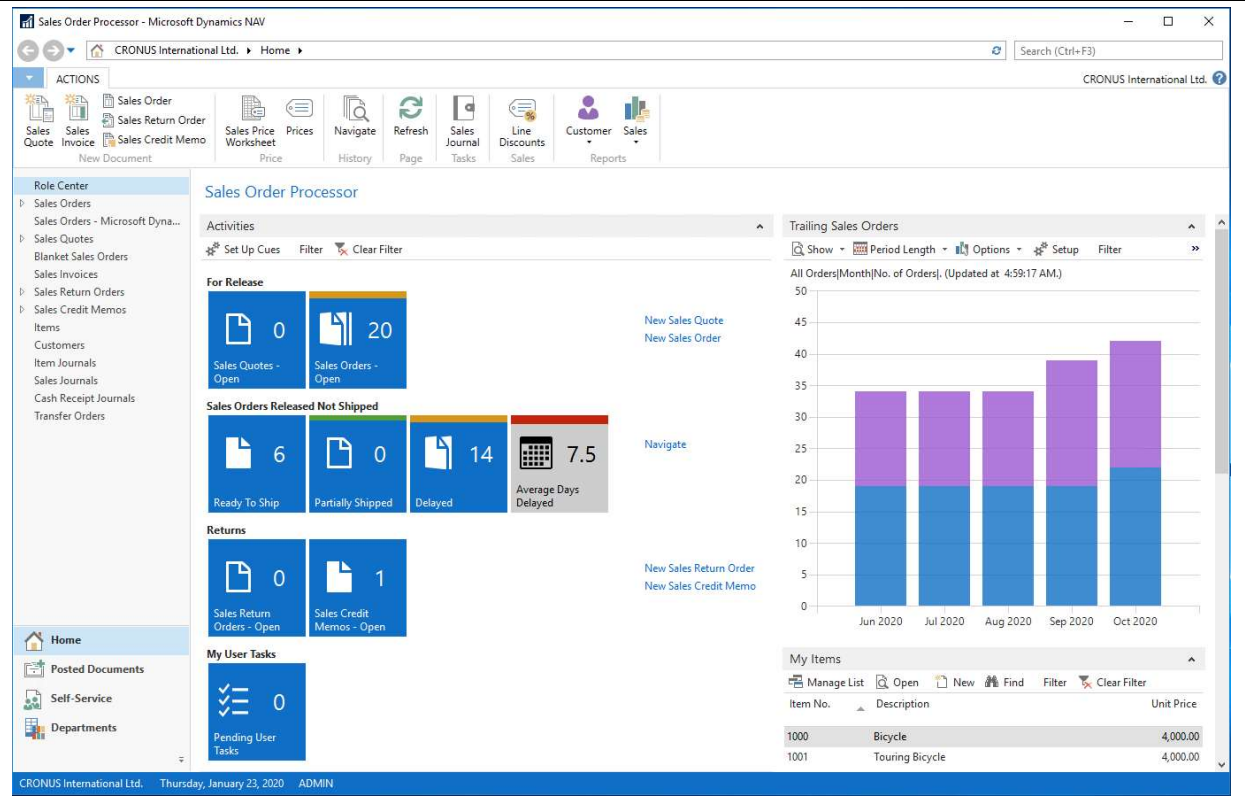
The Windows client is not installed on the Docker host even though it looks like it.

The Docker host shares a folder to the container called **C:\Program Files (x86)\Microsoft Dynamics NAV** – and the container then copies the files from that folder to the host.

This gives the best compatibility and allows the folder to be overridden if deploying a new container.

Use the credentials provided to login.

You can also install the Windows client using ClickOnce. There is a section about this later in the HOL.



### Landing Page

The landing page was the starting point of your journey. You will find all info and links here necessary to connect and use the Workshop VM.

### PowerShell ISE

PowerShell ISE running on the Docker host. This is every IT infrastructure gurus favorite tool and we will be using ISE throughout this Hands On Lab. The NavContainerHelper is installed and ready to use in ISE.

### Visual Studio Code

Visual Studio Code is used for AL development and is not used in this Hands On Lab.

When launching the Workshop VM, the AL Language extension from the landing page is preinstalled. If you deploy a new NAV Container, you will have to uninstall and install a new AL Language extension.

## Basic Docker commands

Let's drill into some of the basic Docker commands to get a better understanding of what Docker is and how it works.

You can run these commands in PowerShell, but Docker is a simple Windows Executable and will run in a command prompt as well.

For simplicity reasons, we will use the Command Prompt.

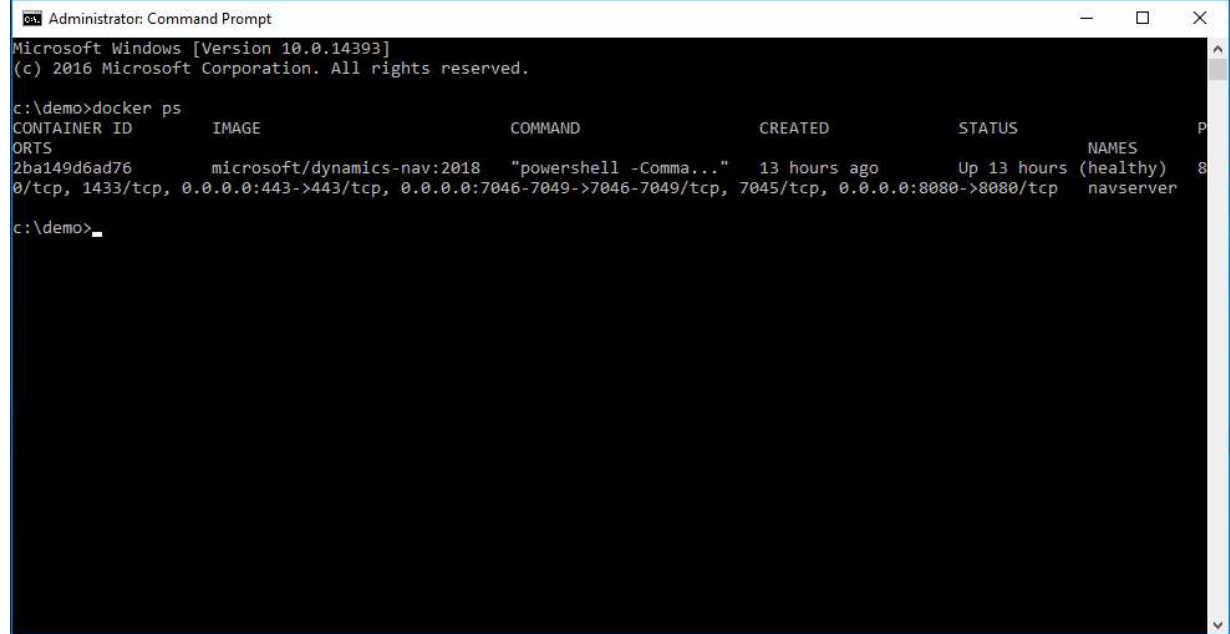
Open the Command Prompt and write:

**docker ps**

This gives you a list of all the running Docker Containers on your machine.

Take some time to Inspect the info:

- The container name is navserver.
- The container ID starts with 2ba149d6ad76.
- The container is based on the *microsoft/dynamics-nav:2018* image.
- Ports 443, 8080 and 7046-7049 are all exposed on the Docker host, meaning you can access them from outside (the internet).
- Ports 80, 1433 and 7045 are open for the host, meaning that you can access them from the host.



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

c:\demo>docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              NAMES
2ba149d6ad76       microsoft/dynamics-nav:2018  "powershell -Comma...  13 hours ago       Up 13 hours (healthy)  navserver
0/tcp, 1433/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:7046-7049->7046-7049/tcp, 7045/tcp, 0.0.0.0:8080->8080/tcp

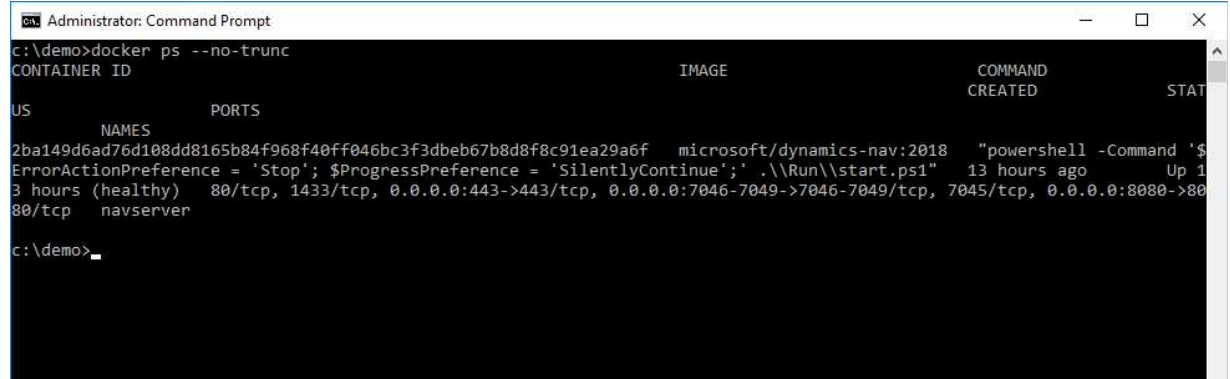
c:\demo>
```

You might wonder why the previous section says: *"The container ID starts with..."*. The reason for this is, that the ID really is a 64 digit globally unique hex identifier, but most time you can refer to the ID by specifying the first digits until your specification isn't ambiguous.

You will get the full ID by typing:

**docker ps --no-trunc**

but if you only have one image you can identify it by writing the first digit – here: **2**



```
Administrator: Command Prompt
c:\demo>docker ps --no-trunc
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              NAMES
2ba149d6ad76d108dd8165b84f968f40ff046bc3f3dbeb67b8d8f8c91ea29a6f  microsoft/dynamics-nav:2018  "powershell -Command '$ErrorActionPreference = 'Stop'; $ProgressPreference = 'SilentlyContinue'; .\\Run\\start.ps1"  13 hours ago       Up 13 hours (healthy)  navserver
80/tcp, 1433/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:7046-7049->7046-7049/tcp, 7045/tcp, 0.0.0.0:8080->8080/tcp

c:\demo>
```



The next command to try is:

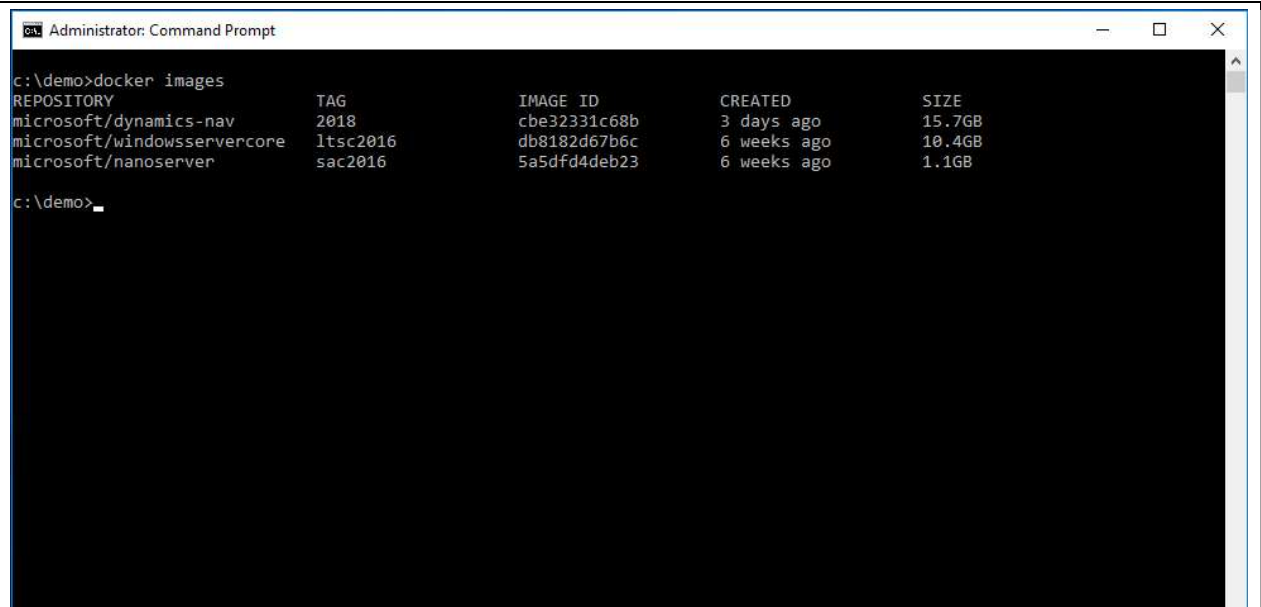
### docker images

This gives you a list of all images available for you to run. In this picture there are the 2 Microsoft base images: Windows Server Core and Nano Server. Beside them, the Microsoft Dynamics NAV 2018 image.

A Docker image is really a set of services installed in a box (container) ready to run on demand.

A specific version of the NAV Docker image is a specific version (incl. localization) of NAV installed in a Container ready to run (ex. NAV 2017 CU7 DK).

The NAV Docker images are highly configurable and customizable.



```
Administrator: Command Prompt
c:\demo>docker images
REPOSITORY          TAG                IMAGE ID           CREATED           SIZE
microsoft/dynamics-nav 2018              cbe32331c68b     3 days ago      15.7GB
microsoft/windowsservercore ltsc2016         db8182d67b6c     6 weeks ago     10.4GB
microsoft/nanoserver   sac2016           5a5dfd4deb23     6 weeks ago     1.1GB
c:\demo>
```

Docker images consists of layers. Layers are shared between images if possible. All NAV images are based on an image called

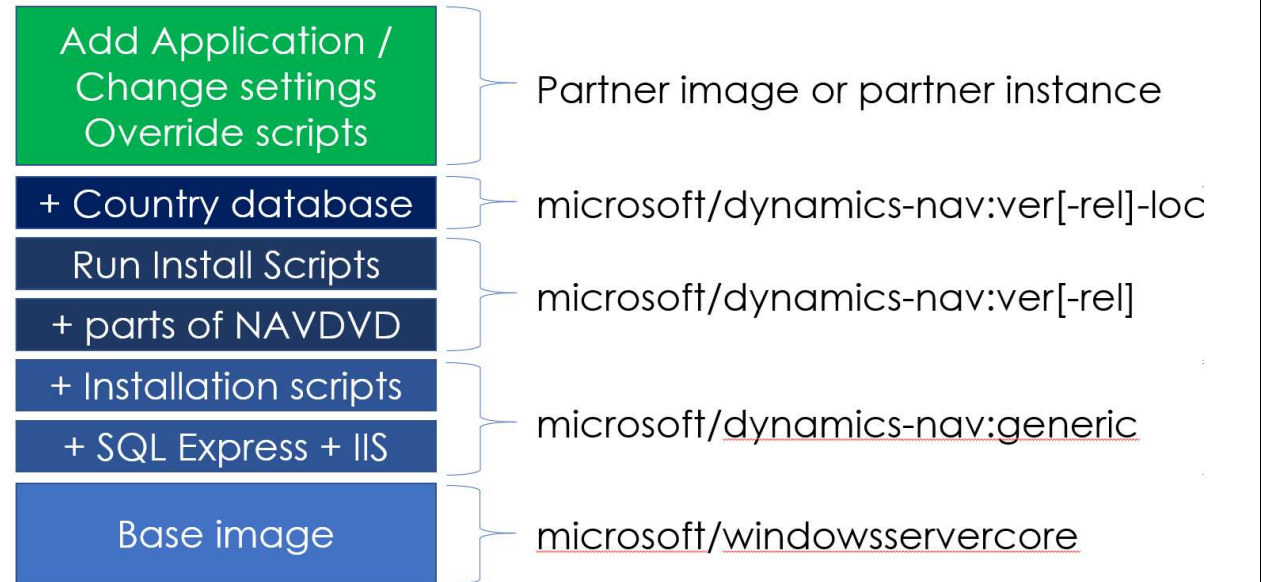
### microsoft/dynamics-nav:generic

The generic image can also be used to run any version of NAV 2013 and up if you have a DVD image. Try to run this command:

### docker pull microsoft/dynamics-nav:2018-dk

You should see that some layers already exists, and the remaining layers are downloaded and extracted.

The extraction typically takes more time than the downloading as the filesystem layer is applied to the existing layers.





All shipped versions of NAV since NAV 2016RTM are available on the public docker hub, where you also can find the EULA and the supported tags:  
<https://hub.docker.com/r/microsoft/dynamics-nav/>.

Under Tags, you will find a list of all the tags in the public repository. Docker images are constructed in layers. That means a Docker pull will only need to download those layers that are different from already downloaded layers.

Docker image names are build up of 3 sections:

<registry>/<repository>:<tag>

<registry> can be a private registry (like **navinsider.azurecr.io**) or the public docker hub, using a single identifier (like **microsoft**).

<repository> is for all NAV images **dynamics-nav**

<tag> determines which NAV image to get.

The tag is build up by this syntax: [[version][cu]][localization]

All parts of the tag are **optional** and if you omit a part, you will get the **latest** (or **w1** for the localization).

Parts are **seperated by a dash** if multiple parts are specified and all parts are specified using **lower case** characters.

Example of valid image names:

**microsoft/dynamics-nav** – gives you the latest cumulative update for the latest NAV version with the worldwide (W1) localization

**microsoft/dynamics-nav:dk** – gives you the latest cumulative update for the latest NAV version with the Danish (DK) localization

**microsoft/dynamics-nav:2017-w1** – gives you the latest cumulative update for NAV 2017 with the worldwide localization

**microsoft/dynamics-nav:2016-cu24** – gives you CU24 NAV 2016 with the worldwide localization

**microsoft/dynamics-nav:2018-cu2-na** – gives you the CU2 for NAV 2018 with the North American (NA) localization

The number of tags is pretty extensive, but you can build up any tag from the above syntax.

The devpreviews are special tags **[devpreview][month][localization]** – devpreview-february is the latest while writing and localization starts with fin (for financials)

Additionally all images are tagged with **[buildnumber][localization]**, where buildnumber is the build number (e.g. **11.0.20348.0**)

As a consequence, **microsoft/dynamics-nav:11.0.20348.0-dk** – gives you NAV 2018 CU2 with Danish localization.

Try:

**docker pull microsoft/dynamics-nav:devpreview-findk**

You will see, that the first ~18 layers already exists and only a few layers will need to be downloaded.

The ~5 layers, which are downloaded is the difference between the US localization and the DK localization, so only this difference will have to be downloaded.

```
Administrator: Command Prompt - docker pull navdocker.azurecr.io/dynamics-nav:devpreview-findk
c:\demo>docker pull navdocker.azurecr.io/dynamics-nav:devpreview-findk
devpreview-findk: Pulling from dynamics-nav
3889bb8d808b: Already exists
9f5eeabe6154: Already exists
ad730affdfb4: Already exists
62dfdb319924: Already exists
f2e764a675be: Already exists
870314b9c01d: Already exists
f18ba3f236bc: Already exists
449f633ba325: Already exists
730b894ed578: Already exists
c663f594832c: Already exists
fe25d21de889: Already exists
c9622587bb72: Already exists
d29f7dc9739c: Already exists
37a0f73207d9: Already exists
deb05332aade: Already exists
0328df32f5e9: Already exists
5d0a8313995d: Already exists
b1c9916bc8d7: Already exists
bd424e2f0540: Extracting [=====] 1.204kB/1.204kB
ed57cb275a9c: Download complete
212012d49b0b: Download complete
5fc4227c641f: Downloading [=====] 58.38MB/155.7MB
45938400664a: Download complete
c:\demo>
```

As you might have guessed by now – if US and DK includes W1 (are built on top of W1), pulling W1 should not cause any downloads. Try:

**docker pull microsoft/dynamics-nav:devpreview**

Indeed – nothing to download, all layers already exist.

```
Administrator: Command Prompt
c:\demo>docker pull navdocker.azurecr.io/dynamics-nav:devpreview
devpreview: Pulling from dynamics-nav
3889bb8d808b: Already exists
9f5eeabe6154: Already exists
ad730affdfb4: Already exists
62dfdb319924: Already exists
f2e764a675be: Already exists
870314b9c01d: Already exists
f18ba3f236bc: Already exists
449f633ba325: Already exists
730b894ed578: Already exists
c663f594832c: Already exists
fe25d21de889: Already exists
c9622587bb72: Already exists
d29f7dc9739c: Already exists
37a0f73207d9: Already exists
deb05332aade: Already exists
0328df32f5e9: Already exists
5d0a8313995d: Already exists
b1c9916bc8d7: Already exists
Digest: sha256:2a08f2cc32948303b83675e6ec858598b538e3ddc9e6b20e4664db5901a20514
Status: Downloaded newer image for navdocker.azurecr.io/dynamics-nav:devpreview
c:\demo>
```

Now, try to run another instance of the dynamics-nav image you have available.

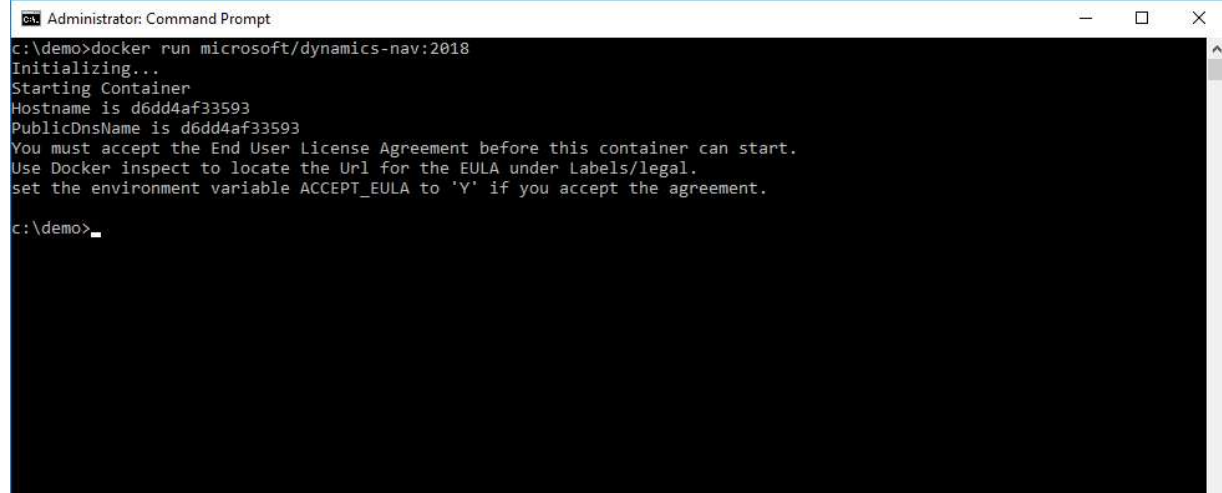
**docker run microsoft/dynamics-nav:2018**

As the error indicates, you will have to accept the End User License Agreement before this container can start.

Use:

**docker inspect --format='{{.Config.Labels.eula}}' microsoft/dynamics-nav:2018**

to get the URL for the legal documents for Microsoft Dynamics NAV 2018.



```
Administrator: Command Prompt
c:\demo>docker run microsoft/dynamics-nav:2018
Initializing...
Starting Container
Hostname is d6dd4af33593
PublicDnsName is d6dd4af33593
You must accept the End User License Agreement before this container can start.
Use Docker inspect to locate the Url for the EULA under Labels/legal.
set the environment variable ACCEPT_EULA to 'Y' if you accept the agreement.

c:\demo>
```

Let's run another instance of the image and accept the EULA:

**docker run -e accept\_eula=Y microsoft/dynamics-nav:2018**

Press Ctrl+C in the command prompt to exit the container and leave it running in the background.

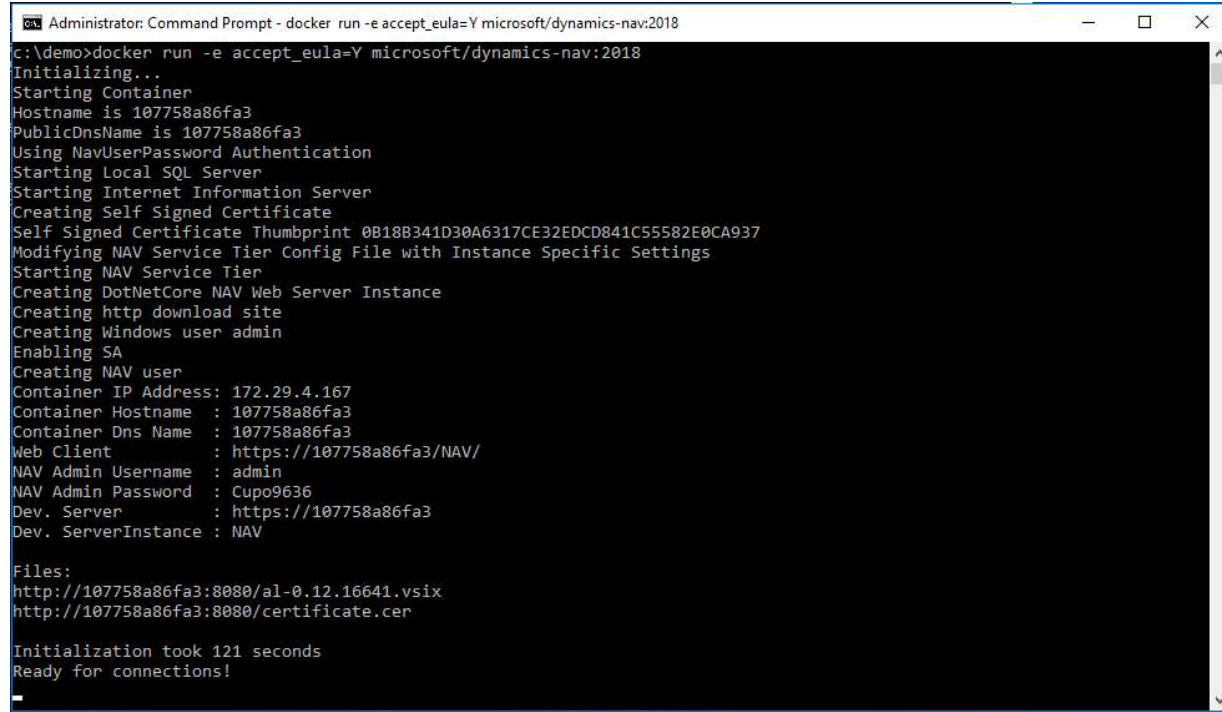
Now, run:

**docker ps**

The command will show you two containers running. Inspect the difference in names, ports etc.

**Note** that Docker automatically assigns a readable name to the container if you don't do so in the Docker run statement.

The original container will have ports exposed on the host, the new container will only have ports exposed on the container.



```
Administrator: Command Prompt - docker run -e accept_eula=Y microsoft/dynamics-nav:2018
c:\demo>docker run -e accept_eula=Y microsoft/dynamics-nav:2018
Initializing...
Starting Container
Hostname is 107758a86fa3
PublicDnsName is 107758a86fa3
Using NavUserPassword Authentication
Starting Local SQL Server
Starting Internet Information Server
Creating Self Signed Certificate
Self Signed Certificate Thumbprint 0B18B341D30A6317CE32EDCD841C55582E0CA937
Modifying NAV Service Tier Config File with Instance Specific Settings
Starting NAV Service Tier
Creating DotNetCore NAV Web Server Instance
Creating http download site
Creating Windows user admin
Enabling SA
Creating NAV user
Container IP Address: 172.29.4.167
Container Hostname : 107758a86fa3
Container Dns Name : 107758a86fa3
Web Client : https://107758a86fa3/NAV/
NAV Admin Username : admin
NAV Admin Password : Cupo9636
Dev. Server : https://107758a86fa3
Dev. ServerInstance : NAV

Files:
http://107758a86fa3:8080/al-0.12.16641.vsix
http://107758a86fa3:8080/certificate.cer

Initialization took 121 seconds
Ready for connections!
```

The docker run command doesn't terminate but will display the output from container directly on the console. The NAV images will, when running, display the output of the event log, meaning that if you do not terminate the console, you will be seeing the event log output in the console.

You can terminate the docker run command by pressing Ctrl+C.

**Note** that the Container will keep running even if you terminate the console.

If you want to run a container without displaying the output, you can use **-d (for daemon)** in the docker run command.

You can always use **docker logs <containerid>** to display the output of the container.

Open the Web client in a browser. Ignore the certificate warnings for the self-signed certificate.

Login with the credentials displayed from the docker run command.

Item No.	Description	Unit Price
1000	... Bicycle	4,000.00
1001	... Touring Bicycle	4,000.00
1100	... Front Wheel	1,000.00



Now try to run

**docker ps -a**

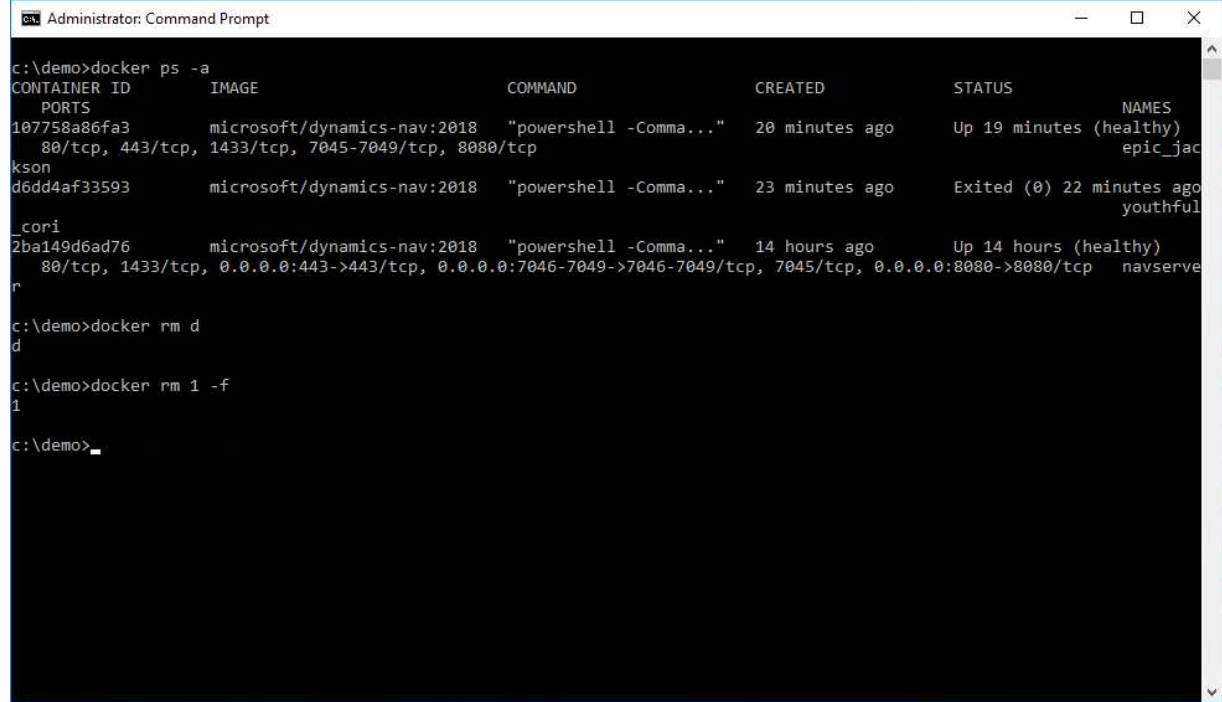
which will show you all containers – running ones and exited ones. If you did try to run a container earlier without specifying the `accept_eula=Y` then you will have an exited container in the list.

Remove the exited container using

**docker rm <containerid>**

If you want to remove a running container you either need to stop it first or use the `-f` parameter:

**docker rm <containerid> -f**



```
Administrator: Command Prompt
c:\demo>docker ps -a
CONTAINER ID        IMAGE               COMMAND              CREATED        STATUS              NAMES
107758a86fa3       microsoft/dynamics-nav:2018 "powershell -Comma... 20 minutes ago Up 19 minutes (healthy) epic_jack
80/tcp, 443/tcp, 1433/tcp, 7045-7049/tcp, 8080/tcp
kson
d6dd4af33593       microsoft/dynamics-nav:2018 "powershell -Comma... 23 minutes ago Exited (0) 22 minutes ago youthful
_cori
2ba149d6ad76       microsoft/dynamics-nav:2018 "powershell -Comma... 14 hours ago   Up 14 hours (healthy) navserver
80/tcp, 1433/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:7046-7049->7046-7049/tcp, 7045/tcp, 0.0.0.0:8080->8080/tcp

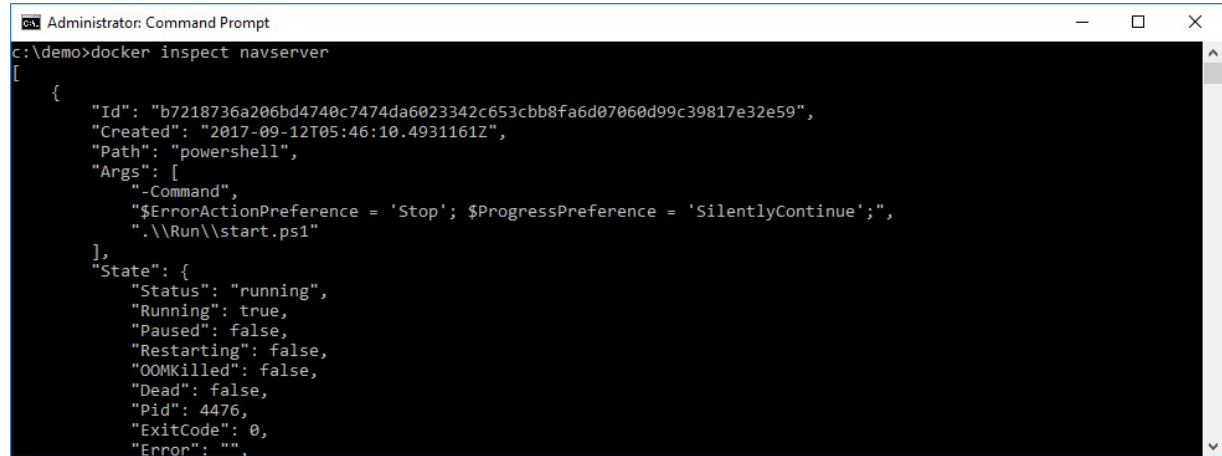
c:\demo>docker rm d
d
c:\demo>docker rm 1 -f
1
c:\demo>
```

Now, let's have a look at the running main container. Try

**docker inspect navserver**

to inspect settings, status, labels etc. on a container or an image.

You will also find network settings etc. if you look through the emitted JSON.

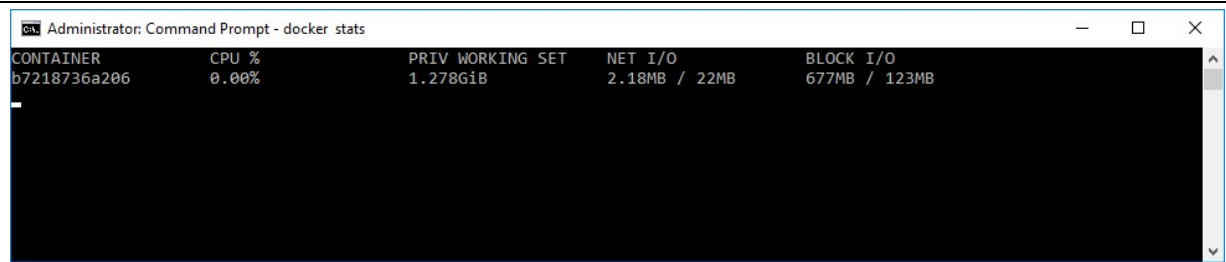


```
Administrator: Command Prompt
c:\demo>docker inspect navserver
[
  {
    "Id": "b7218736a206bd4740c7474da6023342c653cbb8fa6d07060d99c39817e32e59",
    "Created": "2017-09-12T05:46:10.4931161Z",
    "Path": "powershell",
    "Args": [
      "-Command",
      "$ErrorActionPreference = 'Stop'; $ProgressPreference = 'SilentlyContinue';",
      ".\\Run\\start.ps1"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 4476,
      "ExitCode": 0,
      "Error": ""
    }
  }
]
```

Use

### **docker stats**

to get statistics from the currently running containers.



```
Administrator: Command Prompt - docker stats
CONTAINER      CPU %          PRIV WORKING SET  NET I/O          BLOCK I/O
b7218736a206   0.00%         1.278GiB         2.18MB / 22MB   677MB / 123MB
```

If you dislike the format of Docker stats (if you would like the container name included) you can modify the output by specifying a statsFormat property in the c:\users\vmadmin\.docker\config.json file.

In VSCode, create a new file with this content:

```
{
  "statsFormat": "table {{.Name}}\t{{.CPUPerc}}"
}
```

And save it in c:\users\vmadmin\.docker\config.json.

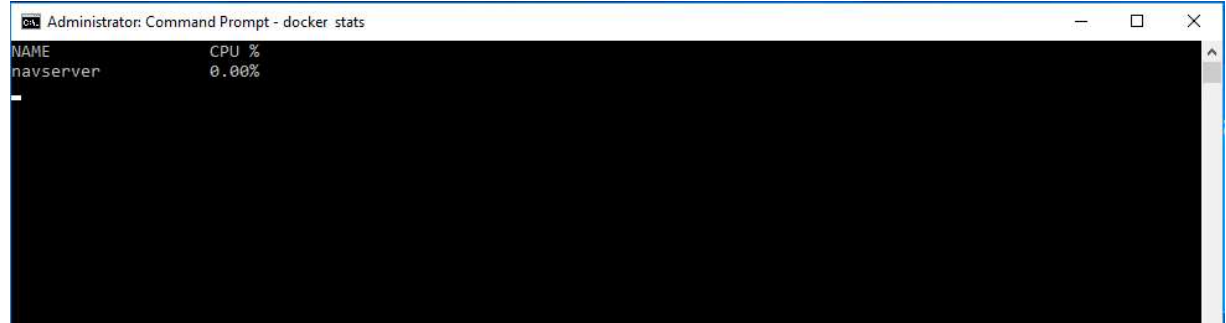
**Note** you need to create the .docker folder in a command prompt using **md .docker**

Now re-run

### **docker stats**

and you will see the info requested.

You can also add a section for psFormat etc.



```
Administrator: Command Prompt - docker stats
NAME          CPU %
navserver     0.00%
```

## Use PowerShell ISE to modify files in the container

If you are new to Docker you might not yet be annoyed over how cumbersome it is to modify files in the container. You can connect using the PowerShell prompt or the command prompt, but since the file system is remote and you don't have a UI, you cannot edit files using Notepad.

But...

You can use ISE – it just requires a small trick.

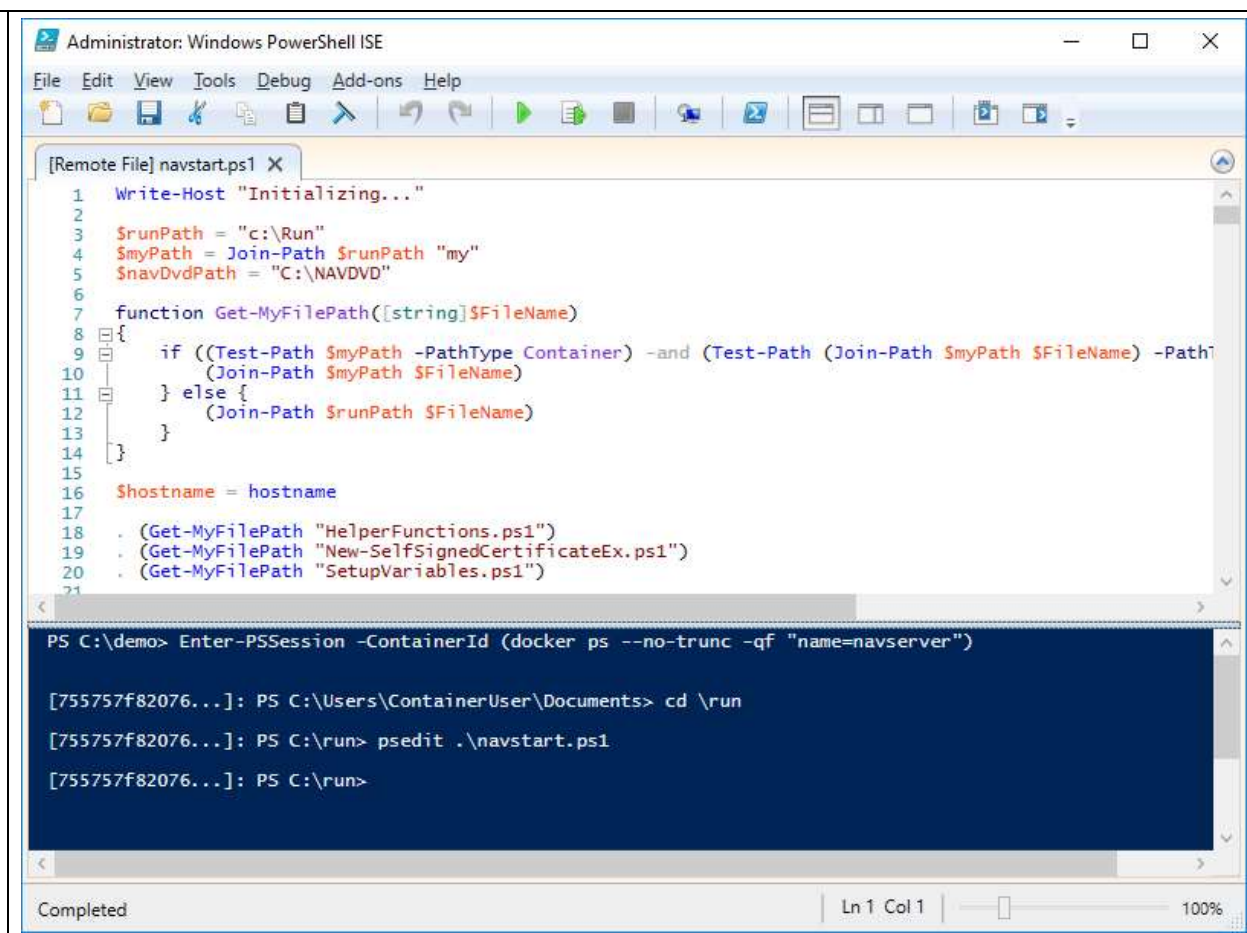
Open ISE and run

**Enter-PSSession -ContainerId (docker ps --no-trunc -qf "name=navserver")**

Now you will enter a remote session in PowerShell (much like the navserver PowerShell Prompt) and inside of this you can use psEdit to edit files remotely without having to share folders and copy back and forth.

PS. The navcontainerhelper introduces a function which is called **Enter-NavContainer <containername>** which does exactly this.

Note that psEdit is an ISE specific function and does NOT work inside the navserver PowerShell Prompt.



```
Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
[Remote File] navstart.ps1 X
1 Write-Host "Initializing..."
2
3 $runPath = "c:\Run"
4 $myPath = Join-Path $runPath "my"
5 $navDvdPath = "C:\NAVDVD"
6
7 function Get-MyFilePath([string]$FileName)
8 {
9     if ((Test-Path $myPath -PathType Container) -and (Test-Path (Join-Path $myPath $FileName) -PathType File))
10        {
11         (Join-Path $myPath $FileName)
12     } else {
13         (Join-Path $runPath $FileName)
14     }
15 }
16 $hostname = hostname
17
18 . (Get-MyFilePath "HelperFunctions.ps1")
19 . (Get-MyFilePath "New-SelfSignedCertificateEx.ps1")
20 . (Get-MyFilePath "SetupVariables.ps1")
21
PS C:\demo> Enter-PSSession -ContainerId (docker ps --no-trunc -qf "name=navserver")
[755757f82076...]: PS C:\Users\ContainerUser\Documents> cd \run
[755757f82076...]: PS C:\run> psedit .\navstart.ps1
[755757f82076...]: PS C:\run>
```

## Advanced parameters

When using Docker run with the NAV image, there are a lot of different parameters you can use. All NAV image specific parameters are specified as environment variables (-e or -env).

There are a number of different parameters you can set when running the NAV Container. This command uses some of them:

```
docker run -e accept_eula=Y -e usessl=N -e auth=Windows -e username=student -e password=<password> --name test microsoft/dynamics-nav:devpreview-finus
```

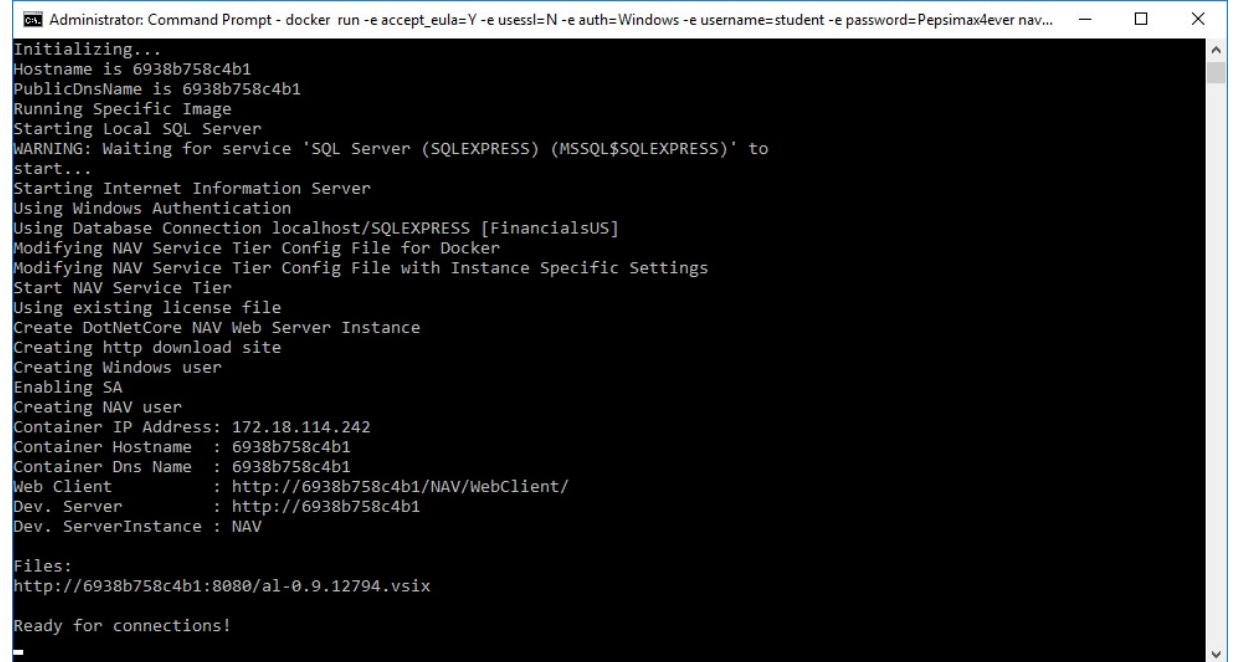
If you specify the password of your student user, then this command will start NAV in a Container without SSL and using Windows Authentication.

Note that this is a known hack, that you can use Windows Authentication between two machines if they share the same username and password.

**Note** that the Web client is now without SSL and if you open it in a browser, you will find that you are logged directly into NAV. Use:

```
docker rm test -f
```

to remove the container named test.



```
Administrator: Command Prompt - docker run -e accept_eula=Y -e usessl=N -e auth=Windows -e username=student -e password=Pepsimax4ever nav...
Initializing...
Hostname is 6938b758c4b1
PublicDnsName is 6938b758c4b1
Running Specific Image
Starting Local SQL Server
WARNING: Waiting for service 'SQL Server (SQLEXPRESS) (MSSQL$SQLEXPRESS)' to start...
Starting Internet Information Server
Using Windows Authentication
Using Database Connection localhost/SQLEXPRESS [FinancialsUS]
Modifying NAV Service Tier Config File for Docker
Modifying NAV Service Tier Config File with Instance Specific Settings
Start NAV Service Tier
Using existing license file
Create DotNetCore NAV Web Server Instance
Creating http download site
Creating Windows user
Enabling SA
Creating NAV user
Container IP Address: 172.18.114.242
Container Hostname : 6938b758c4b1
Container Dns Name : 6938b758c4b1
Web Client : http://6938b758c4b1/NAV/WebClient/
Dev. Server : http://6938b758c4b1
Dev. ServerInstance : NAV

Files:
http://6938b758c4b1:8080/a1-0.9.12794.vsix

Ready for connections!
```



In the above example, test is the container name.  
Most Docker commands take container ID or container name as parameter.

The container name however is not added to the DNS resolver and you cannot ping the container name.

In order to access the container using TCP or HTTP you need to use the hostname.

The default hostname is the first 10 characters of the container ID.

You can specify your own hostname using:

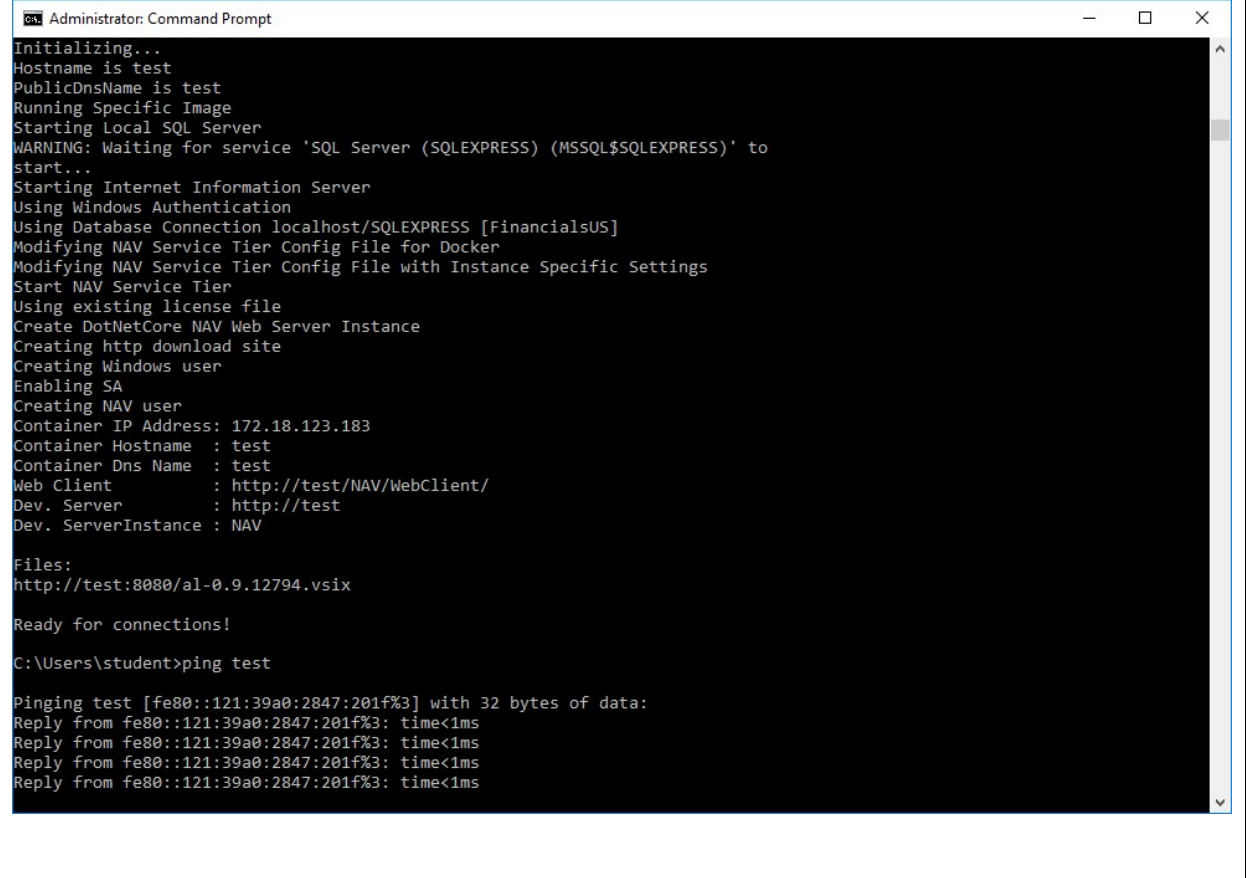
**--hostname test**

Docker will automatically maintain the IP address in the DNS resolution for the hostname, locally on the host.

You can also specify a public DNS name, which is the CNAME record, which points to your host if you are exposing the container to the world using a trusted certificate. PublicDnsName will default to the hostname.

**-e publicDnsName=ws111.navdemo.net**

If you do not use SSL, the publicDnsName is only used for calculating properties like PublicWebBaseUrl, PublicSoapBaseUrl etc. in the config file.



```
Administrator: Command Prompt
Initializing...
Hostname is test
PublicDnsName is test
Running Specific Image
Starting Local SQL Server
WARNING: Waiting for service 'SQL Server (SQLEXPRESS) (MSSQL$SQLEXPRESS)' to
start...
Starting Internet Information Server
Using Windows Authentication
Using Database Connection localhost/SQLEXPRESS [FinancialsUS]
Modifying NAV Service Tier Config File for Docker
Modifying NAV Service Tier Config File with Instance Specific Settings
Start NAV Service Tier
Using existing license file
Create DotNetCore NAV Web Server Instance
Creating http download site
Creating Windows user
Enabling SA
Creating NAV user
Container IP Address: 172.18.123.183
Container Hostname : test
Container Dns Name : test
Web Client : http://test/NAV/WebClient/
Dev. Server : http://test
Dev. ServerInstance : NAV

Files:
http://test:8080/al-0.9.12794.vsix

Ready for connections!

C:\Users\student>ping test

Pinging test [fe80::121:39a0:2847:201f%3] with 32 bytes of data:
Reply from fe80::121:39a0:2847:201f%3: time<1ms
Reply from fe80::121:39a0:2847:201f%3: time<1ms
Reply from fe80::121:39a0:2847:201f%3: time<1ms
Reply from fe80::121:39a0:2847:201f%3: time<1ms
```

## Using the navcontainerhelper

The navcontainerhelper is already installed on the workshop VM, but you can easily install it on your local box from the PowerShell Gallery using:

### Install-module navcontainerhelper -force

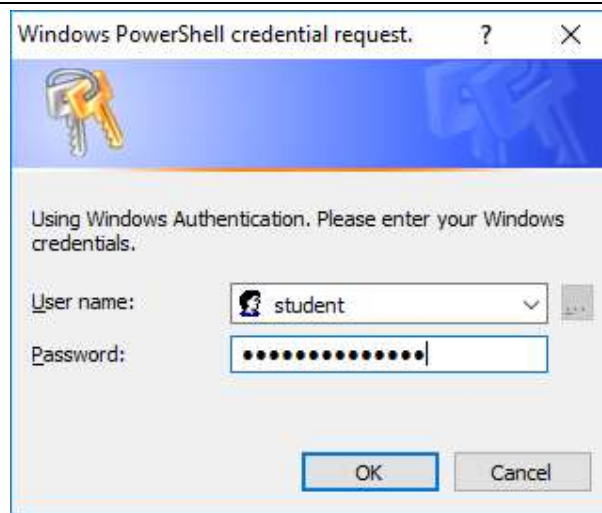
Even though Docker is a command line executable, you can use it in PowerShell like other executables and it does have some advantages. For that, we have created the navcontainerhelper, an open source project which is supposed to make it easier to work with containers.

Create a folder called C:\TEST. Start PowerShell ISE, create a new script and paste in this line:

```
New-NavContainer -accept_eula -containerName  
myserver -includeCSide
```

save it as C:\TEST\start.ps1 and run it.

By default, the New-NavContainer will create a container running Windows Authentication, using the same image as the navserver container. Please supply the Windows Credentials of your workshop VM.



You should see an output, which is like the output on the right here.

Note that first time you run a specific version and include CSide, the container will automatically export all objects as text (baseline for object handling functions). You can avoid this by adding -doNotExportObjectsToText

You can press F5 again and again and the script will automatically remove the old container and start a fresh, this time without exporting objects.

Note that you will have a new set of shortcuts on the desktop to connect to your myserver container.

```
PS C:\Users\freddyk> New-NavContainer -accept_eula -containerName myserver -includeCSide  
-licenseFile C:\ProgramData\NavContainerHelper\license.flf  
Creating Nav container myserver  
Using image microsoft/dynamics-nav:2018  
Using license file C:\ProgramData\NavContainerHelper\license.flf  
NAV Version: 11.0.20783.0-w1  
Generic Tag: 0.0.5.3  
Creating container myserver from image microsoft/dynamics-nav:2018  
waiting for container myserver to be ready  
Initializing...  
Starting Container  
Hostname is myserver  
PublicDnsName is myserver  
Using Windows Authentication  
Starting Local SQL Server  
Starting Internet Information Server  
Modifying NAV Service Tier Config File with Instance Specific Settings  
Starting NAV Service Tier  
Using license file 'c:\run\my\license.flf'  
Import NAV License  
Creating DotNetCore NAV Web Server Instance  
Creating http download site
```

```

Creating windows user freddyk
Setting SA Password and enabling SA
Creating NAV user
Container IP Address: 172.19.145.138
Container Hostname : myserver
Container Dns Name : myserver
Web Client : http://myserver/NAV/
Dev. Server : http://myserver
Dev. ServerInstance : NAV

Files:
http://myserver:8080/a1-0.12.17720.vsix

Initialization took 92 seconds
Ready for connections!
Reading CustomSettings.config from myserver
Creating Desktop Shortcuts for myserver
Export Objects to C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1\objects.txt (container path)
Split C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1\objects.txt to C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1 (container paths)
Export Objects (new syntax) to C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1-newsyntax\objects.txt (container path)
Split C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1-newsyntax\objects.txt to C:\ProgramData\NavContainerHelper\Extensions\Original-11.0.20783.0-w1-newsyntax (container paths)
Nav container myserver successfully created

```

You can use

**help new-navcontainer**

To list all parameters available in the new-navcontainer function.

```

PS C:\Users\freddyk> help New-NavContainer

NAME
    New-NavContainer

SYNOPSIS
    Create or refresh a Nav container

SYNTAX
    New-NavContainer [-accept_eula] [-accept_outdated] [-containerName] <String> [[-imageName] <String>] [[-navDvdPath] <String>] [[-navDvdCountry] <String>] [[-licenseFile] <String>] [[-Credential] <PSCredential>] [[-authenticationEMail] <String>] [[-memoryLimit] <String>] [[-databaseServer] <String>] [[-databaseInstance] <String>] [[-databaseName] <String>] [[-databaseCredential] <PSCredential>] [[-shortcuts] <String>] [-updateHosts] [-useSSL] [-includeCside] [-enableSymbolLoading] [-doNotExportObjectsToText] [-alwaysPull] [-multitenant] [-includeTestToolkit] [[-restart] <String>] [[-auth] <String>] [[-additionalParameters] <String[]>] [[-myScripts] <String[]>] [-CommonParameters]

DESCRIPTION
    Creates a new Nav container based on a Nav Docker Image
    Adds shortcut on the desktop for Web Client and Container PowerShell prompt

```

<p>If you want to spin up a new container with, lets say NAV 2017, you can write:</p> <pre><b>New-NavContainer -accept_eula -containerName nav2017 -imageName microsoft/dynamics-nav:2017 -includeCSide -doNotExportObjectsToText</b></pre> <p>You will see, that the function automatically pulls the NAV 2017 CU12 W1 image and it will take some time to complete the pull as most of the layers are changed.</p> <p>The Generic Tag here is 0.0.5.2 (previous was 0.0.5.3)</p> <p>Again you will find shortcuts on the desktop to connect to your nav 2017 container.</p>	<pre> b3eeb1f92259: Pull complete 95ca09a479f5: Pull complete 77c622d9410e: Pull complete 550c091a6a4b: Pull complete c21730148ab6: Pull complete 25eb60f6a3a2: Pull complete 0b1c5b4248fd: Pull complete 5f0ae6248073: Pull complete e36f63306277: Pull complete b4ee766d2c06: Pull complete bd446c382e13: Pull complete f8fd5c75fab4: Pull complete Digest: sha256:744004a466b6c3550d23c7794f562b4d9b049c0c07cbb7fa43867aac8acf47f Status: Downloaded newer image for microsoft/dynamics-nav:2017 Creating Nav container nav2017 Using image microsoft/dynamics-nav:2017 Using license file c:\programdata\navcontainerhelper\license.flf NAV Version: 10.0.18976.0-w1 Generic Tag: 0.0.5.2 Creating container nav2017 from image microsoft/dynamics-nav:2017 waiting for container nav2017 to be ready, this shouldn't take more than a few minutes Time:          ½              1              ½              2 .....Ready Create Desktop Shortcuts for nav2017 Nav container nav2017 successfully created </pre>
<p><b>Remove-NavContainer nav2017</b></p> <p>Will clean up after your Nav 2017 container</p>	<pre> PS C:\programdata\navcontainerhelper&gt; Remove-NavContainer nav2017 Removing container nav2017 Removing Desktop Shortcuts for container nav2017 Successfully removed container nav2017 </pre>



## Using a different database server

Up until now, we have been using NAV in a Container with the database living inside the same container. That is convenient when doing demos, but frequently you probably want to run the database on a different SQL Server or maybe even on Azure SQL.

The NAV on Docker images have full support for pointing out a different database server, instance, and name on the command line and if your containers are set up with gMSA (Group Managed Service Accounts) and Windows Authentication this should be sufficient to connect.

If you haven't setup gMSA (which is the case with the workshop VMs) you will have to also specify the database credentials.

The below script will create a new container "myserver" running NAV and we'll just reuse the first container "navserver" as a database server. We could also use a new container containing only SQL Server, but we'll be faster this way.

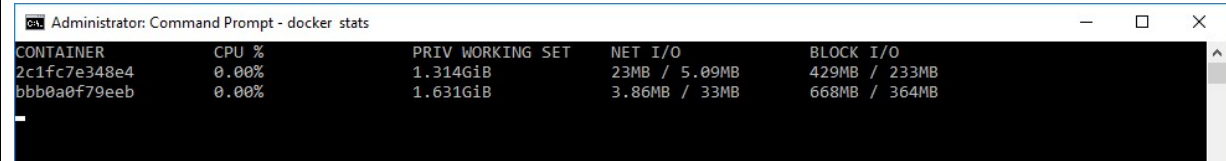
<p>Copy this script, paste it into PowerShell ISE, modify the "&lt;Password&gt;" with the password for your Workshop VM and run it.</p>	<pre>\$password = "workshop4you!" \$securepassword = (ConvertTo-SecureString -String \$password -AsPlainText -Force) \$cred = New-Object System.Management.Automation.PSCredential("student", \$securepassword) \$dbcred = New-Object System.Management.Automation.PSCredential("sa", \$securepassword)  New-NavContainer -accept_eula `                 -containerName myserver `                 -includeCSide `                 -doNotExportObjectsToText `                 -auth NavUserPassword `                 -Credential \$cred `                 -databaseServer navserver `                 -databaseInstance SQLEXPRESS `                 -databaseName FinancialsUS `                 -databaseCredential \$dbcred</pre>
<p>The output from new-navcontainer should be something like this.</p> <p>You will see that the myservers container never starts the local SQL Server, instead it changes the database connection, and imports the encryption key for using a foreign database connection.</p> <p>Try to connect to the navserver Web client and the myservers Web Client (on the Desktop) at the same time and you will see, that they are using the same database.</p>	<pre>Creating Nav container myservers Using image microsoft/dynamics-nav:2018 NAV Version: 11.0.20783.0-w1 Generic Tag: 0.0.5.3 Creating container myservers from image microsoft/dynamics-nav:2018 Waiting for container myservers to be ready Initializing... Starting Container Hostname is myservers PublicDnsName is myservers Using NavUserPassword Authentication Starting Internet Information Server Import Encryption Key Creating Self Signed Certificate Self Signed Certificate Thumbprint 1AEC13120919F358F0C9EBCE4BFC302A3857885D Modifying NAV Service Tier Config File with Instance Specific Settings Starting NAV Service Tier Creating DotNetCore NAV Web Server Instance Creating http download site Creating windows user student Container IP Address: 172.19.147.98 Container Hostname : myservers</pre>

```
Container Dns Name : myserver
Web Client       : http://myserver/NAV/
Dev. Server      : http://myserver
Dev. ServerInstance : NAV

Files:
http://myserver:8080/a1-0.12.17720.vsix

Initialization took 60 seconds
Ready for connections!
Reading CustomSettings.config from myserver
Creating Desktop Shortcuts for myserver
Nav container myserver successfully created
```

Running **docker stats** now reveals two containers and the one running SQL Server and NAV uses more memory than the one running NAV only.



CONTAINER	CPU %	PRIV WORKING SET	NET I/O	BLOCK I/O
2c1fc7e348e4	0.00%	1.314GiB	23MB / 5.09MB	429MB / 233MB
bbb0a0f79eeb	0.00%	1.631GiB	3.86MB / 33MB	668MB / 364MB

## Using the Object Handling Functions in navcontainerhelper

Open the Nav Container Helper prompt and run

```
New-NavContainer -accept_eula -containerName  
myserver -includeCSide
```

To create a CSide development environment next to the navserver container.

Use your Workshop VM credentials when asked for credentials.



Navigate to the C:\ProgramData\navcontainerhelper\Extensions folder and examine the folders:

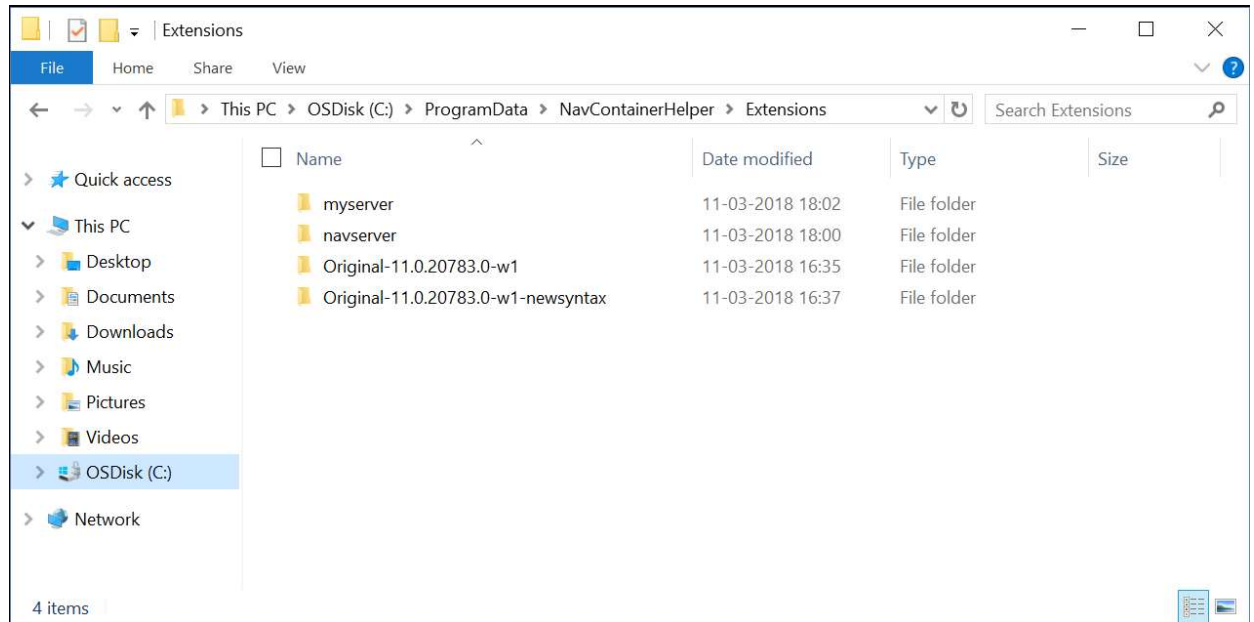
**myserver** is a folder with files specific for the myserver container

**navserver** is a folder with files specific for the navserver container

**Original-11.0.20783.0-W1** contains all the base objects for build 11.0.20783.0 (w1 version)

**Original-11.0.20783.0-W1-newsyntax** contains all the base objects for build 11.0.20783.0 (w1 version) in new syntax format (for txt2al)

The reason for these base object folders are for being able to create deltas from changes in a container.



Start the myserver CSIDE client and modify a few objects.

Please only create modifications which are allowed in extensions v1.

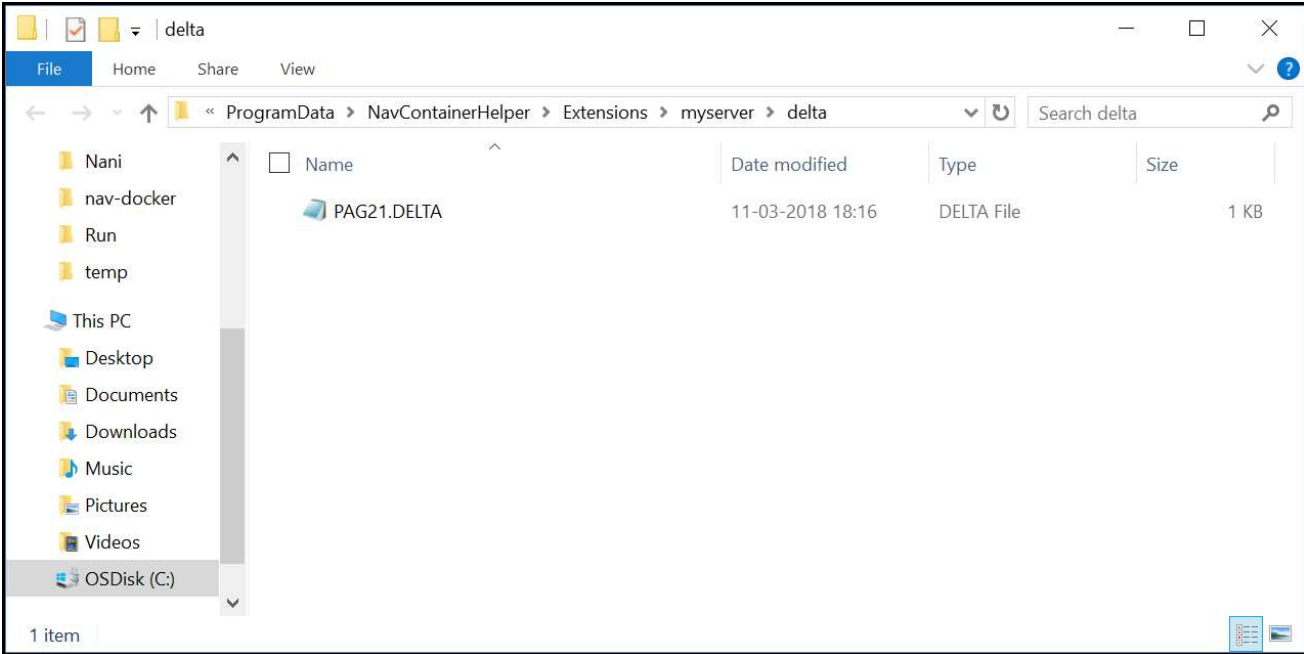
Save your modifications and close the classic development environment.

CRONUS - Microsoft Dynamics NAV Development Environment - [Table 18 Customer - Table Designer]

File Edit View Tools Window Help

E..	Field No.	Field Name	Data Type	Length	Description
<input checked="" type="checkbox"/>	7180	No. of Pstd. Credit Memos	Integer		
<input checked="" type="checkbox"/>	7181	No. of Ship-to Addresses	Integer		
<input checked="" type="checkbox"/>	7182	Bill-To No. of Quotes	Integer		
<input checked="" type="checkbox"/>	7183	Bill-To No. of Blanket Orders	Integer		
<input checked="" type="checkbox"/>	7184	Bill-To No. of Orders	Integer		
<input checked="" type="checkbox"/>	7185	Bill-To No. of Invoices	Integer		
<input checked="" type="checkbox"/>	7186	Bill-To No. of Return Orders	Integer		
<input checked="" type="checkbox"/>	7187	Bill-To No. of Credit Memos	Integer		
<input checked="" type="checkbox"/>	7188	Bill-To No. of Pstd. Shipments	Integer		
<input checked="" type="checkbox"/>	7189	Bill-To No. of Pstd. Invoices	Integer		
<input checked="" type="checkbox"/>	7190	Bill-To No. of Pstd. Return R.	Integer		
<input checked="" type="checkbox"/>	7191	Bill-To No. of Pstd. Cr. Memos	Integer		
<input checked="" type="checkbox"/>	7600	Base Calendar Code	Code	10	
<input checked="" type="checkbox"/>	7601	Copy Sell-to Addr. to Qte From	Option		
<input checked="" type="checkbox"/>	7602	Validate EU Vat Reg. No.	Boolean		
<input checked="" type="checkbox"/>	8000	Id	GUID		
<input checked="" type="checkbox"/>	8001	Currency Id	GUID		
<input checked="" type="checkbox"/>	8002	Payment Terms Id	GUID		
<input checked="" type="checkbox"/>	8003	Shipment Method Id	GUID		
<input checked="" type="checkbox"/>	8004	Payment Method Id	GUID		
<input checked="" type="checkbox"/>	9003	Tax Area ID	GUID		
<input checked="" type="checkbox"/>	9004	Tax Area Display Name	Text	50	
<input checked="" type="checkbox"/>	9005	Contact ID	GUID		
<input checked="" type="checkbox"/>	9006	Contact Graph Id	Text	250	
<input checked="" type="checkbox"/>	50100	test	Text	30	

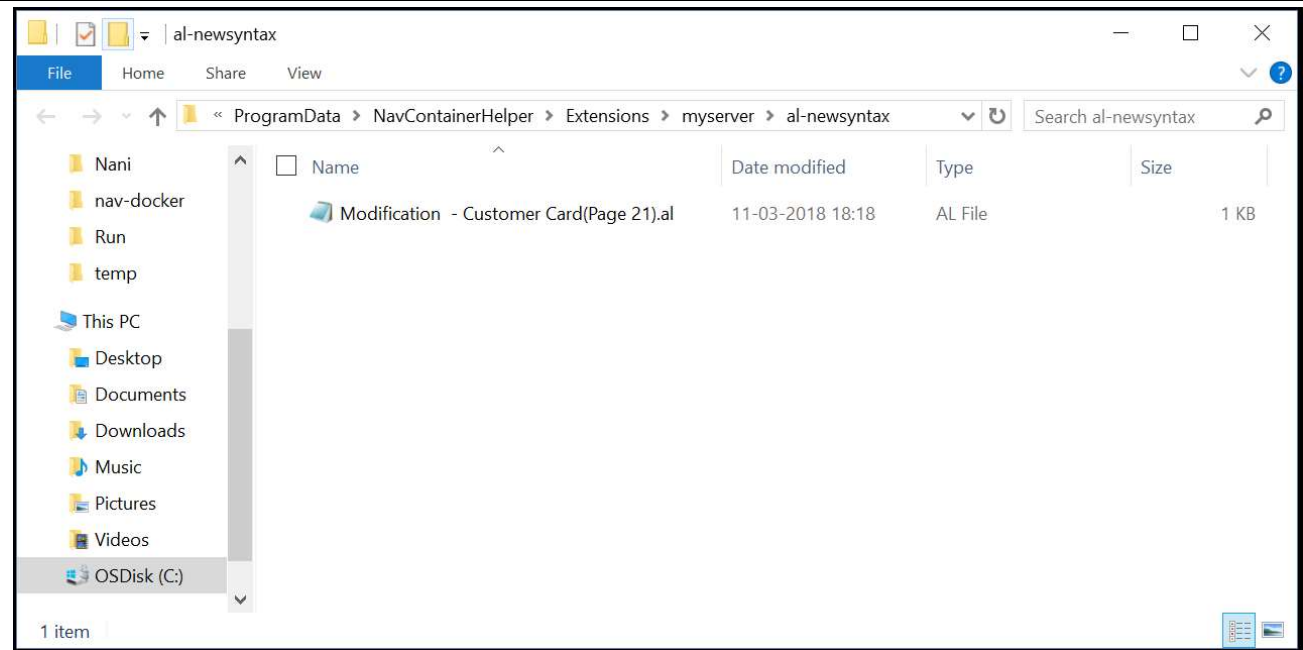
Field Name: test      MYSERVER\student      NEW      INS

<p>Run</p> <p><b>Export-ModifiedObjectsAsDeltas -containerName myserver -openfolder</b></p>	<pre>PS C:\ProgramData\NavContainerHelper\Extensions&gt; Export-ModifiedObjectsAsDeltas -containerName myserver -openfolder Export Objects with filter 'modified=Yes' to C:\ProgramData\NavContainerHelper\Extensions\myserver\modified\objects.txt (container path) Split C:\ProgramData\NavContainerHelper\Extensions\myserver\modified\objects.txt to C:\ProgramData\NavContainerHelper\Extensions\myserver\modified (container path) Copy original objects to C:\ProgramData\NavContainerHelper\Extensions\myserver\original for all objects that are modified (container path) Compare modified objects with original objects in C:\ProgramData\NavContainerHelper\Extensions\myserver\original and create Deltas in C:\ProgramData\NavContainerHelper\Extensions\myserver\delta (container paths) Rename new objects to .TXT delta files created in C:\ProgramData\NavContainerHelper\Extensions\myserver\delta</pre>
<p>You should see a folder being opened with TXT files for new objects and DELTA files for changed.</p> <p>If you navigate to the parent folder, you will find work folders for:</p> <ul style="list-style-type: none"> <li>- original</li> <li>- modified</li> <li>- delta</li> </ul>	
<p>Try also</p> <p><b>Convert-ModifiedObjectsToAI -containerName myserver -openFolder</b></p>	<pre>PS C:\ProgramData\NavContainerHelper\Extensions&gt; Convert-ModifiedObjectsToAI -containerName myserver -openFolder Export Objects with filter 'modified=Yes' (new syntax) to C:\ProgramData\NavContainerHelper\Extensions\myserver\modified-newsyntax\objects.txt (container path) Split C:\ProgramData\NavContainerHelper\Extensions\myserver\modified-newsyntax\objects.txt to C:\ProgramData\NavContainerHelper\Extensions\myserver\modified-newsyntax (container paths) Copy original objects to C:\ProgramData\NavContainerHelper\Extensions\myserver\original- newsyntax for all objects that are modified (container path)</pre>



```
Compare modified objects with original objects in
C:\ProgramData\NavContainerHelper\Extensions\myserver\original-newsyntax and create Deltas in
C:\ProgramData\Nav
ContainerHelper\Extensions\myserver\delta-newsyntax (container paths)
Rename new objects to .TXT
Converting files in C:\ProgramData\NavContainerHelper\Extensions\myserver\delta-newsyntax to .al
files in C:\ProgramData\NavContainerHelper\Extensions\myserver\al
-newsyntax with startId 50100 (container paths)
al files created in C:\ProgramData\NavContainerHelper\Extensions\myserver\al-newsyntax
```

Inspect other object handling functions,  
especially import and compile functions.



## Portainer.io

Portainer is a free GUI for maintaining your Docker environment.

<p>Portainer doesn't work with IE and Edge doesn't run on Windows Server 2016, so we need to download and install Chrome on the Workshop VM from:</p>	<p><a href="https://www.google.com/intl/en/chrome/browser/">https://www.google.com/intl/en/chrome/browser/</a></p>
<p>Copy the PowerShell script, paste it into PowerShell ISE and run it. The script will:</p> <ol style="list-style-type: none"><li>1. Reconfigure Docker daemon</li><li>2. Open port 2375 in the firewall</li><li>3. Create a Portainer directory</li><li>4. Get the IP address</li><li>5. Download and run the Portainer Docker image</li></ol>	<pre>{   "hosts": ["tcp://0.0.0.0:2375", "npipe://"] }   Set-Content "C:\ProgramData\docker\config\daemon.json" restart-service docker  netsh advfirewall firewall add rule name="Docker" dir=in action=allow protocol=TCP localport=2375  new-item -Path "C:\Portainer" -ItemType Directory \$ipAddress = (get-netadapter   Select-Object -First 1   get-netipaddress   ? addressfamily -eq 'IPv4').ipaddress  docker run -d -v C:\Portainer:C\data --name portainer --hostname portainer portainer/portainer -H tcp://\$ipAddress:2375</pre>
<p>Open Google Chrome and navigate to</p> <p><a href="http://portainer:9000">http://portainer:9000</a></p> <p>On your first connection, you will have to create an admin password for Portainer.</p> <p>After that...</p> <p><b>Welcome to a free tool for maintaining your Docker environment</b></p>	